Shopping for CRM Systems

6

Express Version

- Shopping for CRM systems includes creating a long list of candidates, evaluating them against the criteria in the requirements list, and selecting the best two or three for the final evaluation and negotiation process.
- There are so many CRM vendors that it's useful to organize them in categories and to restrict your search to the categories that best match your requirements.

  CRM systems can be mid-range or high-end. Mid-range systems have a good set of functionality and scale fairly well, but they have limited customization capabilities. They are easier and faster to implement than high-end systems. High-end systems have extensive customization capabilities and scale best, but they also require much more time and many more resources to implement. Unless you clearly need the additional functionality and customization tools of high-end solutions, it's best to stay with a mid-range tool.

CRM systems can be suites, covering several business functions, or point solutions that focus on a particular piece of functionality. Although suites deliver pre-integrated solutions, you may need to patch together several point solutions to get the ultimate best-of-breed solution.

Some vendors offer so-called vertical solutions that are customized for specific industries, while most tools are general-purpose. If you need a high-end system you will find that high-end vendors often offer vertical solutions, which should save you some customization work.

CRM systems can be purchased as packaged products or in an ASP arrangement. Some can be purchased either way. If you have reasonably limited customization requirements, need a solution quickly, and do not mind the idea of an ASP arrangement, it may be the best solution for you.

- The traditional RFP process is slow and costly. You may want to substitute a lighter version in which you use the requirements checklist as a scorecard for the vendors.
- Take an assertive approach to driving vendor presentations and demos to minimize fluff and focus on what matters to you—your requirements.
- The key issue in CRM selection is to be perfectly clear about what is part of the product and what is customization. It's often difficult to tell during demos.

Shopping with a Purpose

Once your requirements list is complete, it's time to go shopping. As mentioned earlier, don't start shopping seriously until you have a fairly good idea of your requirements so you don't waste time evaluating unsuitable vendors on the one hand and you don't ignore potential good fits on the other.

It's useful to organize the shopping process into four stages:

- **Creating a long list**. There are hundreds of vendors that claim to be CRM vendors so it makes no sense to attempt to evaluate them all. The first step in the process is therefore to create a so-called long list of likely candidates by performing an abbreviated evaluation of vendors' capabilities against the requirements list.
- **Evaluating the candidates** against the requirements. Through a structured process, you perform a more thorough evaluation of the candidates on the long list against the checklist, rating them as you go.

- **Creating a short list**. Using the results from the evaluations, you narrow down the list of vendors to the two or three best candidates. Creating the short list is usually fairly simple, as leaders emerge rather naturally.
- **Negotiating the best deal**. After the evaluation is complete, you need to check references and negotiate the best possible agreement.

This chapter covers creating the long list, evaluating candidates, and creating the short list. Checking references and negotiating the best deal will be covered in the next chapter.

Creating the Long List

The long list guides the selection process. It should have the following two characteristics:

- Focus on candidates that have a realistic chance of meeting your requirements. For instance, the candidates on the long list should fit within your price range and should offer the high-level functionality you require, whether it's marketing automation or VoIP integration.
- Be diverse enough to include a variety of approaches and philosophies. The long list should not be too short: limiting yourself to a couple of candidates at this stage may cause you to overlook interesting ones. In particular, if you did some vendor browsing as part of creating the requirements list and even if you really liked some of them, you should try as much as possible to consider a wide sample of candidate vendors when you create the long list.

Use a combination of approaches when creating the long list: it's a kind of brainstorm and you want to generate a wide list, so be creative. Below are six different approaches with proven results. Use as many as you can to expand your horizons.

- Visiting the exhibit hall of an appropriate **business conference**, as suggested in the last chapter, is a good way to see many vendors in one go. The level of detail of what you can see in an exhibit hall is ideal for creating the long list—even though it's absolutely insufficient for completing the evaluation step. Seeing systems side by side also allows you to contrast the vendors' positioning easily. The amount of customization is limited because vendors have to address many potential clients, so you are less likely to be confused about what's in the product and what's custom. The experience of seeing that all vendors have slick demos should help immunize you against being taken in by pretty looks during the evaluation phase. Not to mention that you will enjoy collecting the inventive give-aways that are standard fare in that kind of event. (Need another free T-shirt?)

- Study the articles and ads in your favorite **trade magazine or web site**. This is a decent substitute for the trade conference exhibit hall, although the amount of information available in an article is, by necessity, limited while CRM tools are pretty complex. Ads are even shorter than articles but they are interesting because they force vendors to target just one benefit so you will quickly see if the vendor's technical and business vision matches your requirements. See Chapter 11 for suggestions for suitable publications and web sites.
- Sign up for **vendor webinars**. In an hour and without leaving your favorite workstation you can get a high-level picture of a tool. Webinars are often painfully short on exposure to the actual product, devoting half of the typical one-hour length to an "expert" disserting on some lofty topic, another fifteen minutes to a fluffy presentation about the company and its strategic direction, and a scant five minutes to a quick demo. Q&A is fairly short and questions are answered at the discretion of the emcee, so your questions may not be addressed at all. Despite the limitations, webinars are just right for checking on the overall fit. You will find them helpful to build the long list but don't expect to complete a full evaluation through a public webinar.
- Get **suggestions from colleagues**. This is in many ways the best approach since it will yield advice beyond just names. On the other hand, it's limited to your network. It may not uncover the newest tools, depending on your colleagues' appetite for the bleeding edge. If colleagues suggest in-depth evaluations at this stage, politely decline and return when and if their recommendations make it to your long list.
- Get **suggestions from staff members**. Unless your entire staff has been with you forever, it's likely that some staffers have used other tools in other organizations relatively recently. Both positive and negative suggestions are helpful, provided that you take the time to understand the reasons for the enthusiasm or lack thereof (since your needs may be different from the organization they came from) and also the role of the author of the suggestion (since end-user experiences can be totally different from those of administrators or executives).
- Work with **analysts**. CRM analysts are extraordinarily well informed (or should be!) about the CRM world and the relative strengths and weaknesses of the players. In particular, they have a pretty good grasp

of who the stronger vendors are and what vendors may survive for the long haul.

The problem with analysts is that they know nothing about your requirements. The point of CRM tool selection is not to select “the best” or “the leader” in the field, but rather the one that fits your needs most closely. Don’t be mesmerized by quadrants, rankings, or waves; dig deeper into the unique strengths and weaknesses of each offering, matching them to your requirements list.

As you gather suggestions, what is it exactly that you should be considering when deciding whether a particular vendor should be added to the long list? Since you’re not in a position to check each candidate in detail at this stage, stick with a handful of high-level criteria. If the list is getting too long (ten candidates would be a lot), add more criteria from the requirements list. If you can’t find any suitable candidate, drop some of the criteria. The following criteria work well for building the long list:

- **High-level business functionality**. This one is obvious: if you are looking for a marketing automation tool, don’t bother with CRM solutions that don’t offer that. If you need to handle Spanish text, then reject solutions that cannot work with international languages. Make sure you only consider functionality that cannot be customized or somehow added on. For instance, let’s say you want to support chat. While it would be ideal to find a solution that includes chat, you should be able to add chat as a point solution to another tool that provides the other functionality you require. On the other hand, you can’t “add on” international language support or multichannel support.
- **Scalability**. If you are planning for a few dozen users, almost any CRM tool will do. As the number of users increases, the range of tools that can support that many users decreases. Eliminate tools that do not have an established record of supporting twice as many users as you are planning for.
- **Price**. At the long list stage, you are very far from negotiating a final price, but there’s no reason to consider candidates that are way above what you can afford. So if your budget is $100k (for the tool, not the implementation) don’t waste your time on $500k tools, but don’t discard the $200k vendors either, since you can always exercise your negotiating muscles later to get close to or even under your target. We will see in the next chapter that CRM prices can be very flexible.
- **Implementation time**. If you have a short-term implementation goal, some of the more powerful and complex tools are out. Although all vendors claim their tools can be implemented in a few weeks or

months, the more complex tools simply are not amenable to aggressive implementation timeframes. If you need to be up and running in 60 days and the vendor says that they've had implementations as short as 60 days, you probably want to pass on that particular tool unless you are very sure that you can live with a minimal implementation.

Categories of CRM Tools

The CRM field is so wide as to be overwhelming as you enter the long list stage, so it's useful to organize it into a number of categories that meet specific needs. We will look at four useful dichotomies: traditional versus new wave, suite versus point solutions, vertical versus general-purpose, and packaged software versus ASP.

Traditional versus New-Wave Some CRM tools have been around for a while and some are rather recent. The traditional tools have had years to accumulate features so they offer very rich functionality, although the features tend to be piled up in a historical rather than a slick or organized manner. With traditional tools, the architecture and customization tools require longish implementation cycles, but they allow extensive amounts of tailoring if you are willing to undertake the work required.

On the other hand, new-wave tools are streamlined, having adopted a smaller subset of features. The better new-wave tools focus on exactly the essential features, so the loss of functionality may be almost invisible. New-wave tools allow for limited tailoring but on the other hand customization and deployment are easier. Choosing between traditional and new wave tools often boils down to how much customization and integration you need.

Here are characteristics of traditional CRM tools:

- They include **lots of features**. This means that almost anything you may want will be there. For instance, whereas a new-wave CRM tool may lack a CTI integration, a traditional tool will usually offer several options, all of which used by several customers in production settings.

  But lots of features could be a problem. You will probably have to spend significant amounts of time and resources turning off or hiding features that you don't need. And if you choose not to, your users will complain that the application is confusing and hard to use.
- Partly as a result of the abundance of features, and partly because the original thick clients allowed and encouraged it, traditional tools offer **busy screens and can be hard to use** and hard to learn because of the complexity of the underlying application. End-users will typically

need a couple days of training to use the tool properly. This is not a big problem in an environment where staffers use the application a lot and turnover is low, but for staffers who are new or who are intermittent users, the training investment is large.

- Traditional systems **encourage using best practice functionality**, honed over long periods of time. This can be a great incentive to abandon or modify quirky business processes. Actually, if your business model is hard to fit into an existing, traditional tool, your first impulse should be to question your model rather than the tool.

  However, there are instances where the built-in tool workflow simply has to be modified for your needs. Traditional tools make it very difficult to change the workflow, and some changes are simply impossible. For example, collaboration, where multiple individuals contribute to a particular issue, is often desirable from a business perspective. But it's very challenging to build it in an environment that relies on a paradigm of one owner per issue, which is the way most traditional tools function.

- Despite the limitation described above, traditional tools **can be customized to do almost anything—**as long as the workflow model is not tinkered with. This makes them very flexible indeed, to the point where you may not recognize the underlying tool as you check references. Each implementation may not only look but even function in a totally different way.

  The customization possibilities come at a cost. It takes time and resources to customize, for one thing. Another interesting consequence is that the traditional vendors, knowing that almost all customers will perform customizations, adopt a lackadaisical approach to perfecting the out-of the box application and screens, leaving them cluttered and confusing. Worse, they may deliver an "application" that is completely unusable prior to customization; that is, they deliver a tool kit rather than a complete application. If you are in a hurry to get going, a traditional tool may simply not be what you want.

- Traditional tools can also **be integrated to many other systems**. As a bonus you're likely to find reference accounts for many integrations, which is always a nice omen, although not a guarantee that it will work for you too.

  Integrations are costly because there is always some custom work required, so having the potential for integrations may not be such an attractive proposition for you. It's almost always a mistake to select a particular tool over another, simpler one, only because it has integration capabilities that you might exploit in the future but for which you have no specific plans.

- Traditional tools are **expensive and slow to implement**. If you need the richness of their features and their customization capabilities, the cost is probably worthwhile. If you only need something simple they may be a waste of money and resources, and they may saddle your organization with a tool it cannot sustain in the long run.

New-wave vendors take a different approach.

- They usually contain **basic, major functionality**, and are well suited to modest requirements. The better new-wave tools offer an uncannily well chosen subset of functionality: just what you need, and no more. This is much better than having to remove or hide features in a traditional tool.

  As well chosen as the functionality may be, you may find that your needs are much greater than what you can find in a new-wave vendor. If you have to add a lot of functionality that would be bundled in a traditional competitor, it may be easier, cheaper, and even faster to select a traditional tool. As careful as you are with customizations, it's hard to beat the long-term maintainability of built-in features.

- New-wave tools have **less built-in structure**, which can be an advantage if you are trying to implement something different. On the other hand, if you want to enforce a standard best practice, a traditional tool delivers everything you need in a neat package.

- New wave tools sometimes **offer unique features**, both because they come from newer, nimbler companies, and because some new-wave tools are built from the ground up to deliver entirely new functionality. For instance, when chat first became popular for sales and service, a number of vendors appeared that offered pure chat functionality—that is, without a customer repository, the whole focus being on the communication channel. To this day the "chat only" vendors such as divine/eshare have been much more innovative in their domain than the traditional vendors, even the vendors who have incorporated some chat functionality into their products.

  Over time, the most useful and popular new features get integrated into the traditional tools, as chat did, and the inventive new vendors may add more robust CRM features to the new functionality (but chat vendors have not). So if you are looking for unique new features you may have to go with a new-wave solution.

- New-wave tools are **easy to use**, or at least much easier than the traditional tool. This is a consequence of the much smaller feature set to be sure, but the user-interface and the user paradigm are also significantly different. Rather than designing for an expert user who uses the tool daily for years, new-wave vendors design for the user who is computer-savvy but may be new to the application. For instance, they use popular user interface (UI) standards rather than creating an idiosyncratic, complex user experience.
- New-wave tools are, on the whole, **easier to implement than traditional tools**. This is not just because they offer limited functionality, but more a matter of philosophy. Traditional tools are built on the idea that IT help is required and available for tool implementation projects. New-wave tools have a realistic view that IT resources are always limited and they put many tasks within the reach of an educated, but non-technical, business user. Almost all new-wave tools require highly skilled IT resources for the initial implementation, however.
- New-wave tools may have **limited customization facilities**. This is not an issue if you are planning to keep customizations to a minimum and if the base functionality meets your needs by itself. But if you absolutely must have specific functionality that needs to be built expressly for you, new-wave tools may be too restrictive.
- **Integration kits can be missing** or you may find that no customer has implemented the specific integration you are interested in. Here again, your requirements may be such that the limitations do not matter, but if you need specific integrations you will need to carefully evaluate what's available and what's proven.

So should you choose a traditional vendor or a new-wave vendor? Carefully consider your needs for customization and integration. If they are high, chances are that you will be more successful with a traditional tool, but only if you are ready to invest the necessary time and resources to make it work. If your customization needs are modest, or you have limited resources, a new-wave tool that offers a good set of functionality is probably the best bet, even if you have to give up on some less essential features, since the implementation requirements are much lighter.

Suite or Point Solution Another way to look at CRM choices is whether to select a suite offering or a set of specialized tools. A suite offering gives you an integrated solution with different modules that address each business function. A point solution covers one particular area or one particular set of functionality very well, but requires additional tools to deliver a complete solution.

Suite offerings are very tempting.

- They deliver a **pre-integrated solution**, so sales, marketing and service can easily share information. Sharing information is certainly possible with cobbled solutions, but it requires integration work.

  There are a few drawbacks, however. One is that suites do not, in general, solve the issue of integrating with back-end systems such as accounting. The other is more insidious. Whereas point solutions are created with the assumption that they will be integrated with other tools, and hence offer nicely open architectures that conform to industry standards, suites can be weaker in their integration capabilities.
- They usually have a **consistent user interface across functions.** Consistent interfaces make it easier to train users. This may seem immaterial since sales reps rarely morph into service reps or vice-versa, but it does make it easier for reps to research information in other departments.

  Interfaces are not always consistent since some of the suite vendors grew through acquisitions and have not chosen to make the interfaces consistent across functions.
- They usually offer **weaker functionality** in some areas, particularly for the more leading-edge features. If your feature requirements are high, suites may be disappointing.

  In particular, if you are considering adding modules to an existing CRM suite, you may find that a point solution delivers better functionality and is a better choice despite the integration requirements.
- Most suites are offered by **traditional vendors**, so they have the advantages and disadvantages described above.
- Even as traditional tools, suites are **faster and easier to implement** compared to creating custom integrations between best-of-breed tools. If you need a completely integrated solution but are short on time, a suite is your best bet, although you may have to sacrifice some functionality in the process. For the same reason, it may make sense for you to use the same vendor for front-office and back-office functionality.
- Suites are **easier to maintain**, since the maintenance team only needs to learn one tool and only needs to interact with one vendor's support organization. This is tempered by the fact that suites, being traditional tools for the most part, tend to have more complex maintenance tools.

- Point solutions focus on just one area or one particular feature set.
- They deliver the cool functionality and the **specialized or leading edge features**. Some features will eventually be delivered by the suite vendors, but it could take years. Point solutions do what they do extremely well and are often referred to as "best of breed" for that reason.
- With restricted scope, point solutions are usually **easier to implement and to maintain**. But if you need to put together several point solutions, each with a different customization and maintenance environment, what you gain in simplicity may be lost with the synergy of a suite environment.
- Moreover, many, although by no means all, point solutions are also **new-wave tools**, which means their implementation is easier. They also have the drawbacks of new-wave tools.
- If your requirements are narrow, the **limited scope** of the point solutions is not an obstacle. If your requirements are wide and you also need tight integration, then you will need to integrate multiple best-of-breed solutions, a long and resource-intensive process.

Suite or point solution? If you are targeting a single business function, you might as well go for best of breed, at least if you have no plans to expand to other functions in the medium-term. If you need a solution to cover multiple functions, consider both suites and point solutions. If your needs are fairly simple and you are looking for a quick implementation, a suite is your best bet, although you will have to live with the feature limitations.

On the other hand, if your needs are complex and you either are willing to invest in costly integrations or you need only loose integrations, then cobbling together a bunch of point solutions will give you the best of both worlds, high-end features and integration.

Vertical or General-Purpose In an effort to minimize the customization work required to implement their tools, some CRM vendors offer vertical packages that are targeted to a specific industry segment such as health care, financial services, government, etc. Vertical tools are often built with a set of templates applied to a base product, but there are stand-alone vertical tools as well.

If you find a vertical tool with a good fit for your needs you will find that the customization requirements are much reduced. Therefore the implementation time is much faster and the maintainability of the end product is much better. On the negative side, vertical tools are often traditional tools, and therefore harder to implement and to maintain. Mid-range vendors tend to stick with general-purpose tools, so you will probably have to select a high-end vendor if you want a vertical tool. That may be outside your price and timeframe range.

By all means investigate vertical solutions appropriate for your industry, but unless your needs are fairly complex you should be able to do well with a general-purpose tool.

Package or ASP CRM tools have traditionally been sold as licensed software, however some are now packaged as ASP (application software provider) offerings through which the software is rented rather than bought outright. Some vendors offer only ASP solutions, some vendors offer both, and yet others allow third parties to offer the tool in an ASP arrangement while they stick to the licensed software model. Virtually every CRM tool is available through an ASP arrangement. What are the pros and cons of choosing an ASP arrangement?

- **Initial costs are much lower** with an ASP. This is less of a benefit than it seems since you can arrange for a loan arrangement to purchase software. However, if you need a solution only for the short-term, perhaps until you have time to deploy a long-term, comprehensive solution, an ASP will be more advantageous.

  Before you make a decision, run the numbers. Even with an ASP, you will incur significant costs for the initial startup and you may be required to sign up for a lengthy commitment.

- ASP vendors offer pre-packaged customizations, and they have a lot of practice with customizations; therefore **the startup time is remarkably short**. The other side of the coin is that ASP vendors restrict the scope of potential customizations, so you are in a situation that's quite similar to the one of new-wave tools versus traditional tools: startup is easier and faster, but the scope is more limited.

  Be sure to consider the kinds of reports that you need. An ASP can probably create anything that you need, but at a price.

- An ASP solution can **support fluctuations in the user base** more flexibly than an in-house solution. This is true of the infrastructure (no need to purchase or install servers for instance) and of the pricing structure. For instance, if you buy licenses for 100 users and find that you really need 120, you will have to go and buy 20 more, often at a high price because it's not a large purchase. If you only need 80 you can't really return the extra 20. An ASP is able to adapt to ups and downs better.

- In some circumstances it may make sense to **start with an ASP solution and migrate it in-house later** (using the same tool). This is easiest if the ASP happens to also be the vendor. If you are considering such an arrangement, be sure to negotiate the terms for the migration from the ASP model to the licensed model so it has the financial benefits of a rent-to-own situation.
- ASP vendors often offer **prepackaged integrations**, sometimes offering suite-like functionality through integrated best-of-breed tools. This is an ideal situation if you want the best of both worlds, although finding an ASP vendor that offers precisely the set of tools you want can be a small miracle.

  If you want to integrate the system with your existing back-end systems, an ASP solution may prove unworkable.
- ASP vendors offer **streamlined upgrades**, both because they can leverage their experience to many customers and because they tend to support fewer customizations, so upgrades are easier.

  If you want to integrate the system with your existing back-end systems, an ASP solution may prove unworkable.
- ASP solutions are **usually cheaper in the short-term** than licensed solutions, but more expensive in the long run. Make sure you understand all the charges both at startup and in the future, and any back-end charges.
- That being said, ASP solutions **do not exist for low-end packages**. If your budget is very limited, buy a low-end package that will deliver limited functionality, very limited customization capabilities, but a very easy implementation at a low price.
- If you are considering an ASP solution, you will need to consider the important **issues of data ownership, data security, and data migration**. It's a weird feeling to hand overy our precious data, especially customer data, to a third party. You will want to work out ways to ensure that data is regularly backed up and somehow made available to you so that you can continue your operations should there be any issue with the ASP.
- Test and re-test the **performance** of the application if you are considering an ASP. The tool has got to perform well over the network.
- Even more than with a licensed software vendor, double- and triple-check the **stability of the ASP vendor**, since you don't want to find yourself brutally deprived of service.
- You may want to go as far as **designing an exit strategy** if you choose

an ASP. It's comforting to know that you have options should the ASP not work out.

To ASP or not to ASP? The case against ASPs could be as simple as having a corporate policy against them (usually because of data ownership and security concerns). For small and medium companies, however, ASP solutions can be very liberating, allowing the company to focus on its core competency, which is rarely IT. The limits of ASP solutions can actually be quite helpful, since they discourage fanciful customizations that often have limited paybacks.

Packaged solutions are cheaper in the long run, so if you know you will use the solution for a long time, it's best to go that route.

Name, Names, Names!

Naming names in the CRM field is a risky business. The incredible vitality of the field makes for an ever-changing landscape of vendors and tools, and mentioning each and every vendor that exists today would require pages and pages. Therefore the vendor lists below are by necessity partial and there are many perfectly good CRM vendors that are not included. The lists are accurate as of late 2002. For updated lists, please consult www.ftworks.com/JustEnoughCRM.htm.

The best-known names in the CRM world are traditional vendors with high-end offerings such as Siebel, PeopleSoft, and Amdocs. Just to illustrate how common acquisitions are in the CRM field, Siebel offerings include some of the old Scopus line (my ex-employer). PeopleSoft bought Vantive. Amdocs bought Clarify, which was for several years part of Nortel before being spun off. All three vendors offer suites with roots in the front office.

There are relative newcomers to the world of the traditional, high-end vendors such as Oracle, which is trying to leverage its strengths for back-end applications (and the database, of course). Oracle CRM has less of a track record than other vendors, however. Another example is SAP, which could be a good solution if you are already using it for back-office functions.

New-wave high-end offerings include Epiphany—which started in the analytics and marketing automation business and then bought Octane, a service-tracking tool—and JD Edwards, after its acquisition of YOUcentric. Parts of Kana's offerings such as the old Silknet are squarely in the high-end realm, while others are mid-range. All Kana offerings are in the service-tracking arena.

In the mid-range, GoldMine, originally a sales-tracking tool, bought HEAT, a longtime support-tracking tool to deliver a mostly-integrated suite under the FrontRange brand. Onyx, Pivotal, and SalesLogix (now owned by Best Software) also have roots on the sales side but also offer support-tracking functionality. Saratoga Systems is another, smaller player.

The mid-range new-wave field is very exciting, with vendors packing much power into their offerings while allowing for quick and easy implementation—well, at least significantly easier than other CRM tools. Salesforce.com is one example. Salesforce.com is an ASP that started on the sales side, as is obvious from its name, but now also provides basic support-tracking functionality and a decent, if basic, knowledge base. Upshot is a direct rival. And RightNow Web offers a sturdy support-tracking tool with integrated knowledge base and self-service functionality both under an ASP model and a licensed software model.

The big unknown and potentially giant gorilla in the mid-range, new-wave field is Microsoft, which has announced a solution to be available in 2003. The expectation is that the solution will be easy to integrate with other Microsoft tools, including e-mail and perhaps even its accounting offering (Great Plains). It remains to be seen what functionality will be available and how easy the tool will be to implement.

There are also many contenders in specialty areas of the CRM world including collaboration (ePeople, Tightlink), chat (Cisco, divine), knowledge base management (AskJeeves, Banter, Kanisa, Primus), and automated self-service (NativeMinds, ServiceWare) workforce management (Blue Pumpkin), and monitoring tools (Nice, Witness).

The approaches described so far should allow you to create a solid, manageable long list of vendors from which to start the evaluation. It's always possible to remove candidates from the list after you find out that they are lacking some essential feature, or to add a candidate that was discovered late but seems very solid. Experience shows that CRM teams rarely make additions to the long list because their hands are full evaluating the candidates on it, so if you have to err on either side it's best to include more candidates on the long list and be prepared to drop the weaker ones quickly rather than creating an overly short list and missing out on potential good matches.

Evaluating Candidates

The crux of the selection process is the evaluation of the candidates. Evaluating candidates can be extremely time-consuming so it's important to make good strategic and tactical choices on how to conduct the evaluation so you can achieve a good decision in the length of time you allotted to it. The goal of this section is to discuss how to gather all the information that you need in the shortest amount of time.

Keeping Track

Evaluations can be confusing, especially if you are evaluating lots of different vendors. Keeping good records avoids mistakes and saves time, and it doesn't need to take lots of resources. Record keeping is the project manager's responsibility. Since the end goal is to evaluate each aspect of the requirements matrix for each vendor, a good strategy is to summarize all the information in that matrix, keeping supporting arguments arranged by vendor and requirement number. Keep the records centralized so all team members can access them.

To RFP or not to RFP?

The traditional process for evaluating candidates is to use an RFP. The CRM project team prepares a detailed document (the RFP) that defines the tool or service being requested, what kind of information the vendor should provide, and how it should be packaged, and sends the RFP to all the vendors under consideration. The vendors prepare detailed responses to the RFP that are then evaluated by the organization. Based on the results of the evaluation the organization can create a short list of best-fit vendors.

The RFP process is very useful in at least two ways. First, it forces the team to create a complete and detailed requirements list. If you are following the recommendations in this book that won't be a problem but I cannot overemphasize the benefits of knowing what you want before you go shopping. Second, the RFP process naturally creates a formal record of promises made by the vendors. The results of the RFP are usually attached to the sales contract and can be referred to later on if there is a problem with the deliverable. RFPs are almost always used for government contracts and very large contracts for that reason.

If you want to use an RFP process, you can refer to the RFP template at the end of this chapter for inspiration. The template may also be useful if you are not planning to use an RFP: you can use some of the questions as a guide when meeting with the vendors.

Despite its strengths, I find that the RFP process often fails to deliver the benefits one expects, for a number of reasons.

- It takes an **enormous amount of time**. Creating the requirements list is something you should do anyway, so that's not so bad. But the level of detail and completeness you need to achieve to use the list in the RFP is much greater than what you would need for internal use, so it takes much more time than the requirements list itself.

  You also need to give the vendors several weeks to respond to the RFP—and if you give them too little time, the responses will not be as detailed and some of the vendors may decline to participate altogether. The answers also need to be carefully evaluated, including going back to the vendors for clarifications, which again takes weeks.
- It is **expensive**. Tallying up the many hours that are required to create the RFP and to evaluate the answers may shock you. Price tags above $100k are not uncommon for the more complex proposals.
- It can be **misleading**. The accuracy and commitment implied by RFPs seem to guarantee that they will provide correct answers from vendors, but it's not that simple. First, much of the value of the RFP lies in the quality of the requirements. Since CRM systems are complex, it's difficult to word each requirement accurately, succinctly, and unambiguously, and you can't expect vendors' answers to be any more precise than the requirements themselves. Second, CRM systems are very customizable, so vendors may state that a particular functionality is included when in fact it would require some level of customization. The promise of a complete and completely accurate response is not often met.
- Many **vendors actively discourage RFPs**, especially those not at the high end of the market. It's always possible to insist on a response to an RFP, of course, but it requires some persistence and lots of time, which can be in short supply.

Because of the issues with the RFP process, it's often easier and better to use instead a streamlined process through which you actually walk through the requirements list and score each item yourself, based on your experience with each vendor's product. One way of thinking about the streamlined process is that it's very much like the RFP process except that you gather the responses yourself rather than having the vendor write them down before the evaluation. Using a streamlined process has many advantages.

- It can be **quite a bit faster** than the RFP process, since you don't have to prepare a formal document or wait for the vendors to respond (or decipher cryptic answers!)
- It is **likely to yield higher-quality information**. The probability that items will be completely misinterpreted by either the vendor or your RFP evaluation team is much less with a streamlined process. Also, the CRM team is more directly involved in the scoring and so will naturally focus on items of critical interest.
- It is **less resource-intensive**. Not having to create the RFP document is a savings. Although the scoring requires much more work since you need to do it yourself, it's usually easier than evaluating RFP responses. You should find that the streamlined process requires fewer resources, at least if you are shooting for a high-quality evaluation.
- It **allows you to flexibly revise the requirements list** during the evaluation process as you discover new information. Clearly you should strive to have a complete and accurate requirements list right from the start, but it's not unusual to make changes to the requirements list during the evaluation. Managing changes in an RFP process is difficult and confusing.

I very much believe in using a streamlined process but it does have a couple of drawbacks. One, the RFP process forces you to create a very clear and complete list of requirements. If you decide to use a streamlined process instead but you are not disciplined enough to create a strong requirements list, you may end up with a very poor fit because you never bothered to define what a good fit would be.

Another situation in which an RFP process might be preferable is if you need to attach the formal results of the evaluation to the contract. If you want to use your scoring sheet for that purpose you will need to get it approved by the vendor, which will pretty much negate the time savings of the streamlined process. However, you will still benefit from the higher quality of information you obtain by doing the scoring yourself.

Seriously consider using a streamlined process over a standard RFP.

Setting Up Productive Vendor Meetings

Regardless of whether you use an RFP or a streamlined evaluation method, being able to orchestrate productive interactions with vendors will go a long way in shortening the evaluation cycle and, more importantly, in ensuring that you gather accurate and complete information. This important coordination work is usually performed by the project manager, although some delegation is appropriate, at least for larger projects.

To create the long list you might have used conference exhibits and webinars to get a feel for the various tools and vendors. During the evaluation you will want to switch to personal meetings that require coordination and preparation. For high-end tools, the meetings will most probably be handled face-to-face. For the lower-end tools some or all meetings may be held through web meetings and conference calls. The contents and attendance issues are very much the same regardless of the channel, however.

The number and the organization of the vendor meetings depend on the complexity of the project, the size of the team, and logistical considerations. A common approach is to start with an initial meeting of the whole team (perhaps preceded by a smaller-scope evaluation from a smaller group) followed by a series of in-depth meetings attended by only those team members who are interested in the specific area being discussed. For instance, the business users (business owners and super-users) will want to see detailed demos of the functionality. See Figure 6.1 for an overview of the meetings. The balance of this section describes recommended audiences and agendas and specifically describes how to get better demos.

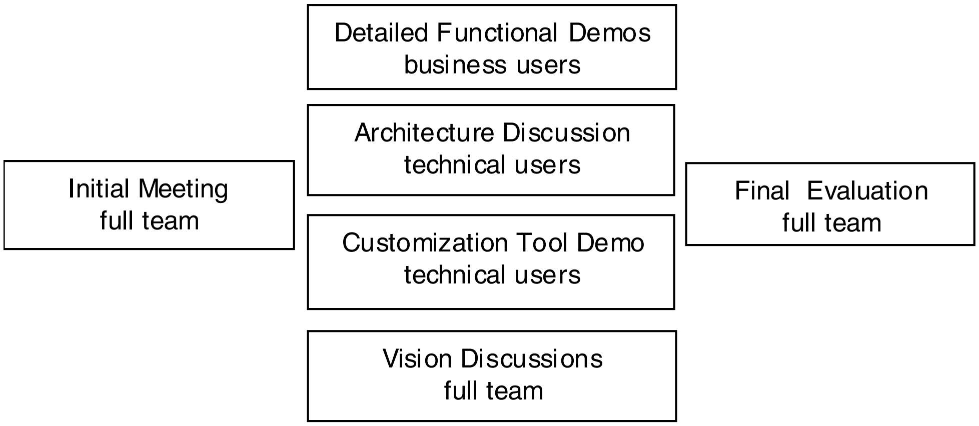

FIGURE 6.1
Vendor Meetings

Who Should Participate? It's very important to hold most vendor meetings with every team member in attendance. It's sometimes desirable or necessary to hold focused meetings for only a part of the team (for instance, an in-depth architectural discussion with only the technical staffers). However, holding separate meetings as a matter of course makes for a disjointed and longer evaluation process, and eventually a poorer decision as each component is evaluated independently of each other.

The project manager must orchestrate the various meetings. The process starts with the introductory meeting, a short meeting with the full team, branching off as needed into specialized meetings, and culminating in a longer group meeting for the final evaluation. The final meeting is almost always held face-to-face, often at the vendor's headquarter so all key staff members on the vendor's side can be present.

Although it's certainly possible to include some team members through teleconferencing while others are physically in the same room, I find that such meetings quickly focus on the participants who are in the room and ignore the remote participants. It's very difficult to stay involved when one cannot hear all the conversation, cannot see what's being demonstrated, or cannot participate easily in the conversation. If you must hold meetings with some but not all participants on a teleconference, take great pains to ensure active participation from the remote participants. Send them materials ahead of time so they can preparefor the meeting and follow along during the discussion. Test the web conferencing connections and other tools that will be used during the discussion. Make a point to include remote participants in the discussions.

If your project team is very large, it makes sense to have a core group do an initial evaluation of the vendors, bringing the entire team into the process only for vendors who pass the initial evaluation. If you choose to use a core group:

- **Make sure all functions are represented** in it. A core group composed entirely of super-users will miss technical architecture items, and a group of technical staffers will miss business functionality.
- **Include different hierarchical levels** in the core group. Don't use the core group idea to shield the business owners and the IT owner (or to shield the super-users and the technical staffers). To ensure the quality of the initial evaluation all levels must be included in the core group.
- Try to **use the same core group to do all the initial evaluations**. This means that it will have the same blind spots (bad) but it also means that it will get more efficient with each evaluation (good) while keeping a good level of consistency (also good).

- **Don't use the core group for the entire process**, only for the initial evaluation. If you feel the core group can indeed handle the whole thing, then perhaps your project team is too large, or not committed enough to the project.

You may think that there is a tremendous burden placed on the project team at this point. Multiple meetings with all the vendors on the long list? That's not often the case. Many vendors can be eliminated after the initial meeting, at least if you use the initial meeting to focus on the critical requirements. Especially with a large project team or a long list with many vendors, a suitable core group should be able to handle the initial evaluation so that the entire team only has to meet multiple times with a handful of vendors.

What Should Be on the Agenda? If you let the vendor set the agenda for the initial meeting, you can expect a rather lengthy presentation on the company, the CRM field, and the high-level product architecture, followed by a brief and slick demo highlighting the more clever features of the product. The demo is usually customized to your requirements to the extent that they are known by the sales rep. A "standard" initial meeting is slick and smooth. It also makes for a terrible way to evaluate the product.

Since standard vendor presentations don't work, you should always set the agenda for vendor meetings. Define both the topics you are interested in and how much time to devote to each. Remember the five categories of requirements?

- Vendor
- Technical architecture
- Functionality
- Implementation and maintenance
- Price

You should aim to cover all five categories in the initial meeting. Of course, you shouldn't bother to dig very much into implementation and maintenance topics until you are satisfied that the tool can deliver the functionality that you need, and you can't expect to get a final quote that early in the process. A typical first meeting can last an hour to an hour and a half (more is better if you want a meaningful demo) and can be structured as follows:

- Vendor introduction—no more than 10 minutes (cut off the vendor after 10 minutes to ensure that you have enough time for the rest).
- Functionality—5 minutes for a short presentation and 20 minutes for a demo. Since twenty minutes is not much for a demo, this is where to

invest additional time if you can stretch the meeting to the recommended 90 minutes. Otherwise, plan a follow-up meeting if you like what you see the first time around. We'll talk more about demos in the next section.

- Technical architecture—10 minutes. This topic will bore the business users and cannot be covered adequately in less than an hour anyway, so you will need a follow-up meeting for the technical staffers but address essential concerns to be able to determine whether it's worth proceeding to the next step.
- Implementation, maintenance, price—5 minutes.
- Q & A—10 minutes.

The project manager needs to spend time with the vendor to prepare for the meeting and to ensure that the vendor fully understands that deviations from the agenda are not acceptable. Otherwise you will find yourself veering back to the standard blend of 95% PowerPoint and 5% canned demo, a most unsatisfying use of the team's time. The project manager also needs to assertively redirect meetings that do not follow the requirements. Better set the stage in the first meeting than suffer through unproductive presentations.

The project manager should conduct a feedback session with the team shortly after the initial meeting. If the meeting is held at the vendor's site, it can be difficult to convene right after the presentation, but try to hold the debriefing very quickly afterwards. The goal of the debriefing is to decide whether to continue evaluating the vendor, as well as to improve future meetings and presentations. Short debriefings should be conducted after each meeting.

If only a core group participates in the initial meeting, assuming that they like what they see, repeat the meeting with the entire team in attendance. You can take advantage of the repeat to further refine the agenda to meet your needs.

After a successful initial meeting with the entire team, you will want to set up follow-up meetings to discuss specific areas. Plan on several follow-up meetings, targeting functionality, architecture, and implementation. The follow-up meetings can be attended by only a subset of the team members. However, it's always desirable to have at least one representative from a non-targeted group in attendance as it helps with information sharing and group decision-making. For instance, ask a super-user to attend the architecture discussion even though it's mostly for the technical staffers. A good format, if you can arrange it, is to schedule parallel meetings for the various subteams and to allow individuals to move from one track to the other for full coverage.

Regardless of how many meetings are scheduled and how they are arranged, they must be planned in advance by the project manager or a team member to ensure that the agenda and attendees are appropriate. One of the main complaints of CRM team members is that they waste time in unproductive meetings. There will be plenty of irritants down the road over which you have little control, so make sure meetings are well organized. Use the outcome of the debriefing sessions to make each meeting more productive than the last one.

Getting a Meaningful Demo Although demos are invaluable to share what the tool can and cannot do, many demos are poorly planned and end up being either divorced from (your) reality or excruciatingly dull as each field and each screen is visited and commented on without an overall vision of how real tasks can be performed. Here are practical ways to turn demos into useful selection tools.

- **Request a vanilla demo**. It's fine if the vendor wants to put your logo on the screen to personalize the tool, but stay away from extensive customizations that obscure the actual implementation requirements. Generally speaking, the more time you give the vendor to prepare the demo the more customizations may creep in, so push for a reasonably close date and make it clear that you do not expect customizations.
- **Define what you want to see** during the demo. An excellent strategy is to go through a normal workflow such as working a customer lead from inception to purchase, or working a customer service case from beginning to end. Stress that you are not interested in seeing each screen sequentially and in detail (you can always do that on the second pass if appropriate), but rather that you want to experience the flow of the product from one task to another.
- During the demo, **ask questions**. If the demo shows corporate customers and your customers are consumers, ask how they can be accommodated. If the routing of service calls is by product and you want to route by geography, ask how you can be able to change that. If your normal workflow requires a manager's approval before sending a quote, ask how that can be built in.
- As much as possible, **drive the demo yourself**. This may not be appropriate on the very first go-around, as you are still getting oriented to the product, but you should definitely do it later on in the process. If the tool is very new, driving the demo allows you to uncover what I politely call soft spots—plainly said, features that do not work yet. Regardless of the maturity of the tool, driving the demo gives you a

very real feel for how intuitive it is. Experienced demo givers make it look effortless to use the tool, and only when you are doing the typing do you realize that the screens are poorly laid out, or that the workflow is not what you need.

- If possible, **get an evaluation copy** of the software to have complete freedom for testing the software. This is now standard with ASPs, since they are all set up for that, but still pretty rare for the licensed package vendors. Vendors are often reluctant to set up evaluations because they fear, often rightly, that evaluations delay the sales cycle while requiring lots of effort on their part. If you decide to use a hands-on evaluation, set a reasonably short schedule and create a formal project plan. Hands-on evaluations are very time-consuming, so eliminate less promising candidates before embarking on them.

  During a hands-on evaluation, ask the super-users to perform the tasks they would normally perform with the software, noting each area of discomfort as well as each hole in functionality. If you have a good workflow defined for the particular business function it should not be too hard to do. For instance, support reps should create cases, work cases, and close cases. Working in a customer role they should be able to use the support portal to open cases, check status, and add to existing cases.

  Meanwhile, the technical staffers can put the software through its technical paces. Is it fast enough? Is it stable enough? Can customizations be accomplished easily?

  Often the evaluation copy will be installed at your site by the vendor. Assign appropriate technical staff to be present during the installation to see how difficult it is. In addition, the vendor staffers that are doing the installation are quite open and much can be learned through informal conversations with them. The smart project manager may want to hang out in the data center on the day of the installation and chat them up.

Tough Questions for CRM Vendors

Cleverly arranged demos are very powerful, and so are pointed questions to the vendor. I list here questions about the technical and functional aspects of the product. Pricing-specific questions are covered in Chapter 7, “Buying CRM Systems,” and questions about integrators are covered in Chapter 8. For best effect, ask the tough questions several times during the evaluation from different individuals and compare the responses you get for consistency.

- Are you using your own tools in-house?

  All vendors have customers. If the vendor is not using its own tools in-house, something's very fishy.

  Ask for a demo of the way the application is used in-house. Does it look anything like what you saw in the demo? Usually the in-house version is kept pretty close to the vanilla version, both because it makes it easier for upgrades and also because the cobbler's children are poorly shod. Question any differences between the customer demo and the in-house demo. The demo giver is often a regular employee, not a sales rep, so you have another opportunity to get candid input.

  Is the version used the current version? If not, why not? Using older versions internally is a sign that upgrades are painful. Make a note of it.

  Is the tool integrated with any other tools within the organization? If not, it may be yet another manifestation of the cobbler's children getting poor footwear, or it could be that integrations are complex and difficult. Try to find out why.

  Ask how many people are responsible for administering the system and what their skills are. Also find out how many users the system supports and what problems they are encountering with it.

- Is it vanilla or is it custom?

  This question should be repeated for each and every feature of the product. A most useful question during demos, it can also be refined to "What work is required to create this particular customization?" Some tools have easy-to-use facilities to create minor customizations, so that minor changes are not an issue.

- What happens to customizations when you upgrade?

  Upgrades are often very difficult for CRM tools because customizations need to be examined one by one to decide whether they are still needed for the new version. Then, the ones that need to remain need to be re-implemented against the new version. Grill the vendor on this topic as much as possible and request recommendations for creating customizations that minimize the maintenance requirements in the future. Questions about how to handle customizations during upgrades should be on your list when you talk to references as well.

  Another good question about upgrades is to ask the percentage of customers who are running the latest release. Unfortunately, there's no way to audit the answer, but the number of customers running the new release is a good indication of how difficult it is to upgrade versus how compelling new features are.

- Do you have a user group?

  A vendor without a user group would be rather suspect to me. Ask whether the user group is driven by the vendor or is independent. Most user groups receive significant financial and operational assistance from the vendor, which is certainly a good investment in terms of customer satisfaction and marketing.

  If there is a user group, take time to talk to its representatives, keeping in mind that often the people who are active in the user group are great fans of the product.

- Can we talk to your support manager?

  Speaking as an ex-support manager, I know that all the dirty laundry eventually gets aired to that group. While the support manager will be restrained when talking with a prospect, much interesting information can be gleaned from seemingly innocent questions. Ask how many support staffers are in place today, what kind of profile they have, and what the current backlog of cases is compared to the incoming volume. (Remember that CRM systems are complex, so a backlog of about two weeks worth of cases is normal—but much more may mean that problems are hard to troubleshoot). You should also ask what percentage of cases are bug-related (ask this question of multiple individuals—more than 10% indicates that the product is overly buggy).

- What key features are you planning to release over the next year?

  While you must refrain from buying a system based on future features, which may or may not be released, and may or may not be released on schedule, it's very useful to understand the short-term product priorities. Do they fit with your priorities? If not, then perhaps another vendor would provide a better fit.

Another source of relevant questions is the RFP template at the end of this chapter. The questions are valid whether or not you are planning to use a formal RFP process.

Creating the Short List

Having asked the right questions, having seen the right demos, you have the tools to create the short list. The short list should be really short, with only two or three vendors, and should consist only of tools that you are happy with from a technical and functional perspective. It is often the case, in my experience, that you are truly happy with only *one* vendor, but force the team to consider a backup so that you are not forced into an unhealthy financial surrender to one vendor. On the other hand, if you have lots of vendors that meet your needs at this point, look a little harder at which ones are the best fit from a functional perspective.

It's a Gradual Process

Creating a short list is rarely a big-bang event. Gradually, as the evaluation unfolds, candidates may fall by the wayside, sometimes as early as the very first meeting when the key requirements are evaluated: too expensive, or not robust enough, and off they go.

Don't be afraid to disqualify candidates as you discover major issues with them so you can focus on more promising ones. You can always go back to them later if the front-runners fail to fulfill other requirements.

Scoring the Requirements

Since there are so many factors to selecting a CRM tool, it's useful to have a logical process to analyze your findings. Whether you are using an RFP process or a streamlined process, it's very useful to create some kind of a rating matrix for the various candidates to organize the scoring.

Use the Requirements List If you did your homework for the requirements list, you have the essential elements for a rating matrix: the requirements list. It's fine to add, delete, and change some requirements as the evaluation progresses, but if you find yourself making significant changes you should go back to creating a long list again.

Define Weights

Not all requirements are created equal, so it makes sense to give weights to the various elements in the requirements list. Rather than spending hours assigning very precise weights, I suggest you use a simple 1/2/3 weight selection, starting with assigning each element a weight of 1 (normal, lowest weight) and picking out the key requirements to have a higher weight. For instance, the must-have requirements can have a weight of 3 and all others a weight of 1.

Don't spend too much time fiddling with the weights. I've found that most evaluations end with remarkably few strong candidates so that the decision hinges on strategic considerations rather than a few points here and there, weights or no weights.

Score Each Item Here's the fun part. Go through the entire requirements list and score each vendor still in competition for each requirement. This needs to be done whether or not you use an RFP (in other words, don't just take the vendor's word for meeting a requirement). It is useful here to compare vendors to develop a robust scoring method. For instance, if you are scoring scalability and you need the tool to support 500 distributed users, you may want to give 10 points to vendors who have multiple production installations with more than 500 distributed users, versus 5 to vendors that only have one such installation.

Scoring is often an iterative process. Taking the scalability example, you may realize as you are scoring that you did not confirm the exact number of users for each reference so would have to go back to the reference before completing the scoring. This is completely normal and should be planned for in the project schedule. I like to start the scoring process relatively early so I can spot and correct problems before the scheduled end of the selection phase.

Add 'em Up Unless you are spreadsheet-challenged, adding up the scores should be pretty easy. Once it's done, compare the scores. Typically the candidates that the team thinks are best come out with the best scores (although not always in the order one would expect, as we will discuss below) and the others come out significantly behind.

If you find large surprises, such as an underdog coming up with great scores, go back and analyze the areas that made the difference. It could be that that the weights for the scoring system are not defined appropriately, in which case you can go back and fix them. Another reason for surprises is that the team's impression of a vendor is strongly colored positively or negatively by the relationship with the sales team. If that's the case, remember that the sales team will fade away as the purchase is completed. If the tool is poor you will be stuck with a poor tool anyway. If the tool is great but the sales team is difficult to deal with, make an effort to work with other individuals on the vendor's side, in particular the post-sales team: are they efficient and friendly? That's more important than the performance of the sales team in the long term.

On or Off the Short List?

The whole business of scoring is to help you make a decision, but scores cannot and should not make the decision for you. Compare the scores but also trust your intuition: if a candidate is scoring higher but the team truly likes another one better, it may well be that the preferred candidate is the better one for you. The scoring sheet is only a tool and it may not be perfect, usually because of the choice of the weights. If a candidate truly feels better than another, it's probably the better choice.

Thank the vendors that did not make it to the short list (making sure that you have at least one backup to your preferred candidate). There's no need for them to expend more energy at this point, or for you to put effort into maintaining a relationship with them. As you say "no thanks" you may be surprised to receive some interesting financial proposals from the rejected vendors. If the only reason for rejecting a particular vendor was that the expected price tag widely exceeded your budget but you are now offered something reasonable, by all means reverse your decision.

Sample RFP

This section contains a sample RFP structure that you can customize if you wish to use an RFP process. RFPs are massive documents because each and every item in the requirements list will need to be translated into an RFP question. This sample simply refers to the requirements list when appropriate to avoid needless repetitions, but shows the other sections in detail. Many companies have standards for RFPs so use the suggestions here to augment the process that already exists in your organization if there is one.

RFPs are organized as follows:

- A cover page
- A cover letter
- RFP instructions
- Company information
- Vendor qualifications
- Product overview
- Technical requirements
- Functional requirements
- Implementation and maintenance requirements
- Pricing information include configurations, often spreadsheet
- Appendices: vendor exhibits

Let's explore what each section contains.

Cover Letter

The cover letter is simply a transmittal memo that summarizes the purpose of the RFP, orients the vendor to the way the RFP is packaged, and states the basic requirements for responding to the RFP.

Date

Vendor Contact
Vendor Name
Vendor Address

Dear Vendor

We have selected you as a candidate vendor to provide a proposal for a CRM application.

This package contains complete instructions for preparing and sending the completed proposal. Please follow the instructions exactly so we can easily evaluate your responses as well as the other vendors' in an objective manner. I am available to answer any questions that may arise. I can be reached by e-mail or by phone.

Please return your completed proposal by Date. We will be unable to consider proposals received after that date.

Sincerely,

Project Manager Name
Organization Name
Phone Number

RFP Instructions

This section describes how the RFP process will work and gives specific instructions on how to respond to the proposal.

Proposal Guidelines Start by giving a short summary of the context of the RFP: why are you interested in a CRM tool? What are you trying to achieve? What target dates do you have in mind? This is only a summary so the vendor can confirm its interest in preparing a response. A more detailed description is included in the next section. Then, give the following factual information about the process:

- Contacts and communications (on your side). Who will be able to help the vendors with questions?

- Evaluation and selection process: how will the RFP process be run and what criteria will be used to perform the evaluation?
- RFP and selection schedule. This is not a formal commitment to making a decision by a particular date, but it's useful for the vendor to understand your general timeframe. RFPs for CRM systems rarely exceed a couple of months for preparing the answer, and another month or two for reaching a decision.
- Effective dates of pricing: give yourself plenty of time in case the selection process is delayed.
- Legal considerations for the RFP. Ask your legal team to confirm what is best to include here. Almost always there is a so-called "right to reject" clause, stating that you are free to make a decision on any basis, including deciding not to make any purchase at all, with no recourse possible for the vendors.
- Confidentiality: it is customary to ask the vendors to keep the RFP confidential and to commit to keeping the answers confidential except as required to make a decision.
- Costs: vendors should bear all costs of responding to the RFP.

Vendor Instructions Give instructions to the vendor on how to complete the RFPs, typically to provide clear and concise answers, to use the rating scorecards for the technical and functional requirement sections, and perhaps to provide pricing in a spreadsheet format. State that the RFP answer will become part of the contract.

Give detailed instructions on when to respond and how. Electronic responses are common and they are often preferred because they can be shared easily. State any format requirements.

Company Information

This section gives the vendors information about your organization. It's important because it sets the tone for the proposal and it allows the vendors to tailor the RFP to your particular needs. The company information section should include the following subsections:

- A company profile including the market you serve and the goals of the business (one or two pages).
- Your existing technical environment including corporate standards, applications, database management systems, workstations, network topology, and vendors for all of the systems. Information about how

the systems are administered is not always given, but I find it very helpful to include it as well.

- A description of the business functions to be served by the CRM tool. For each business function, include its specific role within the organization ("Partner Sales" may be obvious to you, but maybe not to the vendor), its staffing level and reporting structure, the locations, goals, measurements, and anything else that pertains to how the system should work. Define whether the tool is replacing an existing tool, and if so why. Include your business goals for the new system. This section can be several pages long for a complex CRM project.

Vendor Qualifications

This section asks the vendors to describe their corporate fitness to meet the requirements. Many of the questions are taken from the "Vendor Requirements" section of the requirements list. Many deserve the full written answers that are asked since they don't lend themselves to the scoring system we will see later for the technical and functional requirements.

Vendor Profile This section typically starts with a free-form statement from the vendor. I recommend limiting the length here for fear of getting back a very long and not very relevant expose. It is useful to get a feel for the history of the company, though, as it can indicate much about the vision and the executive team. After the overall statement, questions in this section include:

- Business vision
- Financial history (typically you want to request the financial statements for the last two or three years)
- An overview of the products offered (limit the length here again!)
- Industry experience: does the vendor have similar clients?
- R&D investment: how much is the vendor investing in developing new products and new technologies
- Geographical location of development, sales, and support offices
- Partnerships with relevant vendors
- User group
- ISO 9000 compliance
- Product awards

Product strategy Ask the vendors to explain their plans for new releases, new markets, and new customers. Place a limit on how many pages you will consider or ask specific questions to avoid getting massive amounts of information that's not directly useful. I recommend asking both for a long-term plan (say 3-5 years) and for the contents of the next major release, which should be firmer.

References Specify type of customer, location, configuration, etc. We'll discuss references more fully in Chapter 7. Ask the vendor to provide contact information for each reference.

Contractual Terms An RFP is not the place to negotiate detailed contract terms, so your goal is to gather information on standard terms so you can get ready to negotiate. If specific terms and conditions are important to you, by all means include them in the RFP.

Specifically request:

- The terms of a standard contract
- Acceptance criteria
- Warranty information
- Non-disclosure protections
- Payment terms
- If the vendor may provide implementation services, terms and conditions for the implementation, including a statement of work, ownership of the finished product, and warranties on the work and on the milestones.

Product Overview

Provide an overview of the recommended system.

Technical Requirements

The technical requirements section, like the Functionality Requirements section that follows it, is usually constructed to match exactly the requirements list you created. Both require the vendor to answer each question in detail as well as rate its ability to match the requirements in a scorecard similar to the one we showed in the last chapter when we discussed requirements.

Questions take time to create, which is one of the main reasons why the RFP process is resource-intensive—the other reason is the scoring—but they are often necessary to communicate the nuances of each requirement. Compare the following RFP question:

> Describe your technical architecture and how it supports scalable deployments. If several options are available, describe the ones that are better able to meet our requirements as discussed in the statement of work section.

with its corresponding, terse item from the requirements list:

> Scalable architecture

It's clear that you will get more information from the question than the item as is. And, at the same time, it will take you much longer to interpret and score the answer to the question than to look at the rating. Whenever possible, use short and precise questions to increase your chances of getting unbiased information.

Start the section with instructions on rating the items. It's usually very dangerous to use a simple yes/no rating system, since anything that's remotely possible will be rated with a "yes". Most RFPs use a system similar to this one with four categories:

- Features that are available today in the out-of-the-box product
- Features available in future releases (the vendor should specific both the release name and its targeted date)
- Features that require minor modifications (no coding)
- Features that require major modifications (coding)

Functional Requirements

This section is by far the longest in the RFP, matching its length in the requirements list. Proceed as you would for the technical requirements section, asking vendors to provide both written answers and ratings.

Implementation and Support Requirements

This section explores the implementation and support requirements per the requirements list you created. If you expect the vendor to provide implementation services, you will probably have to create a separate statement of work and negotiate it separately, but you can ask about high-level information in the product RFP. Even if you are completely certain that the vendor will not be the integrator, it's useful to get the vendor's high-level estimates of what an implementation would consist of so you can use it when selecting an integrator.

Here are items to request for the implementation:

- Overview of implementation project
- Technical requirements
- Sample project plan with schedule and deliverables
- Strategies for each major phase: planning, requirements gathering, testing, rollout
- Suggested staffing including qualifications and roles
- Recommended strategies for minimizing maintenance requirements of customized tools

For training, request the course list, complete course descriptions, pricing, end-user training options, and the availability of a train-the-trainer program. Request recommended backgrounds and custom curriculum suggested for system administrators and implementers. Request training schedules and course availability: if courses are always booked, your technical staffers may need to wait for a long time before they can be useful.

You may want to request sample user documentation in the RFP.

For support and maintenance, be sure to detail the requirements down to the SLA (service-level agreement) level.

Pricing

Request firm pricing quotes for the configuration(s) you require. It's useful to request several quotes for different configurations, as it may be advantageous to purchase a license for slightly more users than you currently have. Quotes should include implementation (if provided by the vendor), maintenance, and other services such as training.

Request future pricing information as well. For instance, are maintenance price increases capped? How will maintenance be computed in the future?