

DEVISING AN EFFECTIVE OUTSOURCING STRATEGY

EVALUATING AND MANAGING THE OPTIMAL R&D OUTSOURCING
MIX

This paper is for R&D organizations and R&D managers who are just beginning to embark upon an outsourcing strategy, as well as for those who have tried some form of outsourcing but have been disappointed in the results.

It details a variety of characteristics that can be used to define a software project in terms of the outsourcing strategy that will best suit it. A simple, repeatable model for revealing a specific project's outsourcing profile is also provided – a model that you can use for every software project that you manage.

Macadamian
software@macadamian.com
1-877-779-6336

700 Industrial Ave.
Ottawa, ON Canada K1G 0Y9
www.macadamian.com

THE “VERTICAL” EVOLUTION

A common evolution that occurs in all research and development (R&D)-driven industries – such as pharmaceutical, electronic, automotive – is now changing how the software sector operates. In an emerging industry, organizations typically begin life as horizontally integrated entities; that is, they perform all functions internally, from innovation to manufacturing, information technology (IT), administration and finance. In a new industry where processes and best practices have yet to be defined, R&D organizations must maintain tight controls over all aspects of the business.

As the industry matures, however, the trend toward vertical integration accelerates. A number of factors contribute to this shift, including:

- Margins tighten, so companies look for ways to do more with less. They come under pressure to increase margins and contain costs. One way of doing so is by offloading non-core aspects of the business to third parties that focus on those areas.
- Technologies, equipment and other supporting infrastructure become commodities, causing companies to place greater focus on honing, marketing and selling their core competencies – the intellectual property that truly differentiates them in their markets.
- Processes are refined and best practices are developed. This knowledge becomes widely available, making it feasible for firms to specialize in particular process areas. Because the processes are well known and expectations are clear, R&D organizations can effectively manage third-parties performing the specialized work.

As the software industry enters its maturity, the trend toward vertical integration is accelerating. In 2004, 25% of software companies polled said that they outsource some part of their software development; in 2005 that number was closer to 35%, and continues to grow¹. In fact, every large R&D-driven organization today has an outsourcing strategy, and the most mature of these use a variety of flavors in their outsourcing mix.

This paper is for those R&D organizations and R&D managers who are just beginning to embark upon an outsourcing strategy, as well as for those who have tried some form of outsourcing but have been disappointed in the results. The paper details a variety of characteristics that can be used to define a software project in terms of the outsourcing strategy that will best suit it, taking into account all flavors of outsourcing, including:

- Offshore
- Nearshore/onshore
- Local contracting
- In-house development

The paper also provides a simple, repeatable model for revealing a specific project's outsourcing profile – a model that you can use for every software project that you manage.

¹ Software Development Magazine

THE THREE DAMNING MYTHS OF OUTSOURCING

Before we consider a model for defining software projects in terms of the optimal outsourcing strategy, we must first dispel three common myths about outsourcing. In almost every situation where software outsourcing has not been successful, one or more of these myths was at play as an assumption made by the R&D organization.

1. Outsourcing means offshoring

Outsourcing refers to delegating a particular task, project, or area of responsibility to an entity outside of your own organization. The third-party entity may be a sub-contractor in the same city (or even the same office), a specialized firm on the same continent, or a development house overseas. Too often, however, “outsourcing” and “offshoring” are used interchangeably, and in so doing, the full range of outsourcing options is overlooked. This paper addresses all of these options and provides a model for determining how and when to mix them for optimal results.

2. Offshoring is cheaper

This myth goes hand-in-hand with Myth #1. By confusing outsourcing with offshoring and missing out on other options, companies may send the wrong types of projects to offshore development firms under the assumption that, “you can’t go wrong with \$14.00/hour labor.” As this paper will explain, however, that approach is often prohibitively expensive because offshoring the wrong kind of software project is counterproductive and inefficient. Often, an “onshore” development team with higher hourly rates will propose a lower overall project cost because they can ramp up in a matter of days, rather than weeks.

3. Outsourcing means “over the wall”

A common misconception surrounding outsourcing is that it allows an R&D organization to throw work “over the wall” and forget about it until a deliverable is returned. But, that’s simply not the case, as legions of failed outsourcing projects of all sizes have proven. Outsourcing requires at least as much management as an internal project; depending on the type of project, different kinds of management will be necessary. This paper helps you to understand what type of management will be required for each kind of project and outsourcing model – from fully hands-on management to nearly complete automation – and the range of scenarios in between.

PROFILING YOUR SOFTWARE DEVELOPMENT PROJECT

Outsourcing a software development project does not have to be a crapshoot. Developing a mature and effective strategy is made simpler with a repeatable process – a model – for evaluating the type of project at hand and matching the outsourcing mix to that model.

The simple but highly effective model presented here is based on software project management best practices; specifically, evaluating your project based on the “3Cs”: **Communication**, **Coordination**, and **Control**.

Characteristics of your *project* will influence the type of partner you choose.

Characteristics of your *company* will influence the type of partner you choose.

Evaluating how much of these your project has or will require is the basis for the evaluation model presented later. First, each of the 3Cs is defined.

Communication

How much communication will your project require? This includes both instant and planned, synchronous communication between you and the project leader, the project leader and the development team, and between technical contacts within the team.

Note that communication in this sense does not refer to communicating status or directing work; that is dealt with later under “Coordination.” At this point, consider:

- How much instant communication will be required at each stage in the development lifecycle to explain the design, the intent, and the big picture?
- How much will you need to explain “why”; for example, why you chose a certain architecture, why the product will have a certain set of features, and so on?
- How often will you need to adjust or make changes to the design direction?
- Does your process dictate a lot of instant communication, or does your company culture place a high value on planned communication?

In a software project, the amount of communication required will be determined by product maturity and the processes you have in place.

Product Maturity

Is the software product in its early stages (concept, alpha, beta, etc.), is it a stable product (version 1.0 and later), or is it a mature product (version 6.0 and later)?

In early stages, a product requires the development team to innovate. This involves many unknowns, requires experimentation, and therefore a great deal of communication – much of it instant communication held “at the water cooler” or via chat, email and *ad hoc* phone calls. In the advanced-concept phase, where ideas are being tested with customers and rapid prototyping is underway, requirements may change weekly or daily, also requiring frequent communication.

In early stages, a product requires the development team to innovate, which implies many unknowns, experimentation, and communication.

The more mature a product is, the fewer unknowns there are. Most of the development in a mature product (typically version 6.0 and up) is sustaining development and fixing bugs. The product is well defined, and developers can see the product running and determine how it works. The need for ongoing communication in mature product development is low because much can be understood by reading the code and the documentation.

Processes in Place

When considering the processes you have in place, think of the type of culture in your organization as well as the type of project at hand. Does your team favor frequent *ad hoc* communication, or is your process more deterministic – with scheduled meetings, formal design and code reviews, and planned interactions?

Consider how your team normally operates. Do you follow Agile methods or Waterfall methods? An Agile process implies frequent, continuous communication, while a Waterfall process relies less on communication and more on documentation and planning.

Rate Your Project

Keeping the above considerations in mind, on a scale of 0-5, rate the level of **Communication** your project needs:

0	1	2	3	4	5
0 = minimal communication is necessary			5 = frequent instant communication is necessary		

Coordination

Coordination involves directing the work and following up to make sure a project is on track. How much coordination will your project require to be a success? Several factors will determine this:

- Time
- Culture
- Availability of resources
- Potential for change
- Process maturity

Time

Time-sensitive projects require a high degree of coordination. Short-term projects, such as early version product development or customizations for a key customer, are higher risk because there is no time to manage slippage or to add resources to meet a deadline. Someone must be on top of the project at all times.

Culture

If your development team is dispersed around the globe, details like differing holidays, time zones and language will impact coordination. The more dispersed the team, the more coordination will be necessary.

Availability of Resources

Often, a new project begins with the intention of having a dedicated resource to oversee it, but that resource is often a trusted senior developer or manager who is likely to be called away to solve a critical issue. Be realistic about what internal resources you can actually dedicate to coordinating your project.

Potential for Change

New products and early-stage products come with a high likelihood of change. For mature products, significant changes should be treated as new projects – for example, the porting of a mature product to a new platform.

Process Maturity

Software development process maturity can be quantified by the amount of process and/or automation in place for source control, bug tracking, and so on. Where tools are not available, processes for manual coordination will be required.

Rate Your Project

Keeping the above considerations in mind, on a scale of 0-5, rate the level of **Coordination** your project needs:

0	1	2	3	4	5
0 = no coordination is necessary			5 = a great deal of coordination will be required		

Control

Control encompasses the tools and processes used to control a software project, namely the configuration management, defect tracking, development processes, and especially project tracking tools. When evaluating a project's control, consider both what level of control will be *required*, as well as what control infrastructure is already *in place*.

Tools

The continuum between early-stage and mature products can again be applied here. If the product is mature and the maintenance required is primarily bug fixing, then most of the visibility that you will require into that project can be provided by tools. The bug tracking system, configuration management and project management tools will tell you 80% of what you need to know about the project's status.

An early-stage or time-sensitive project will not be as easily controlled and tracked using tools. Hands-on, human intervention will be necessary.

An early-stage or time-sensitive project will not be as easily controlled and tracked using tools. More hands-on, human intervention will be necessary. If your software project falls into this category, consider what percentage of the project can realistically be controlled through automation. As a rule of thumb, the standard development tool set can oversee only about 20% of the information management in a project that requires a high degree of innovation.

Infrastructure

The amount of infrastructure and process you have in place will also determine what level of control a project requires. The more sophisticated your configuration management and defect tracking system is the less manual process coordination will be necessary. Consider these aspects of your control infrastructure:

- Can it be securely extended to remote teams?
- Can it report on important metrics such as code churn, bugs reported vs. bugs fixed and project velocity.
- Do you have an automated project system in place through which development tasks can be assigned, managed and tracked?
- Does it pull together development and project management metrics to give you and your managers a dashboard view of the status of the project?

Rate Your Project

Keeping the above considerations in mind, on a scale of 0-5, rate the level of **Control** your project needs:

0	1	2	3	4	5
0 = full toolset is in place			5 = project cannot be controlled by tools		

Next, you'll plot your project on our simple modeling tool >>

PLOT YOUR PROJECT

It's now time to plot your project onto a modeling map. This will help you to easily interpret your project in outsourcing terms. Using the blank models provided in Appendix A on page 14, plot one software project per model.

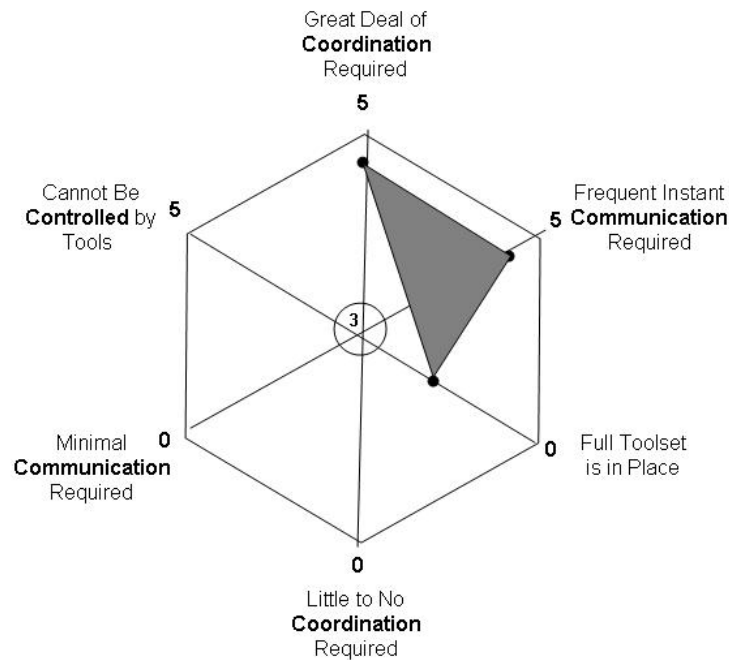
How-Tos

To plot your unique project model, refer back to the score (from 0 to 5) that you gave to each of the three factors described previously: Communication, Coordination, and Control.

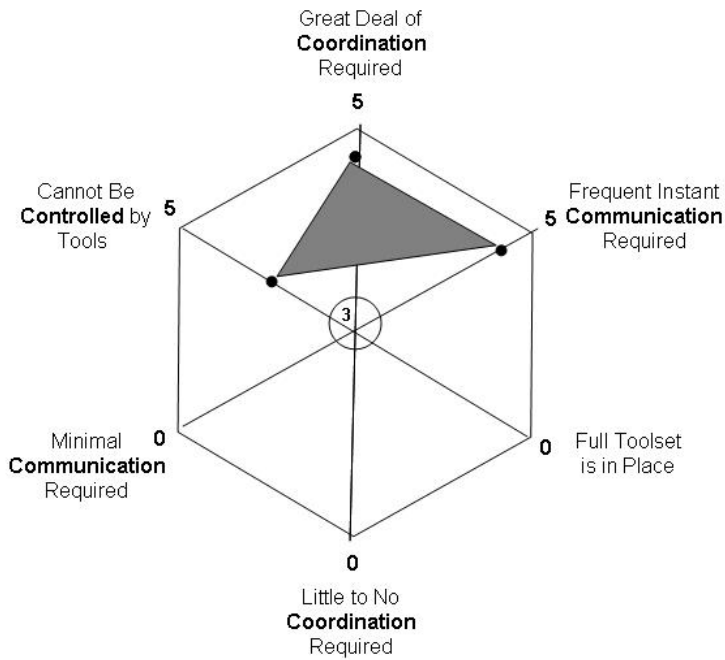
The model includes three (3) distinct axis, each representing one of the three Cs. Begin with Communication. Find the "Communication" axis on the modeling map, then place a dot along that axis to represent the communication score that you gave your project.

Repeat this process for Coordination and Control. Then, draw a line between the dots – the resulting shape should be some form of triangle. Once your project is plotted, read on to understand the best outsourcing model for the project's unique profile.

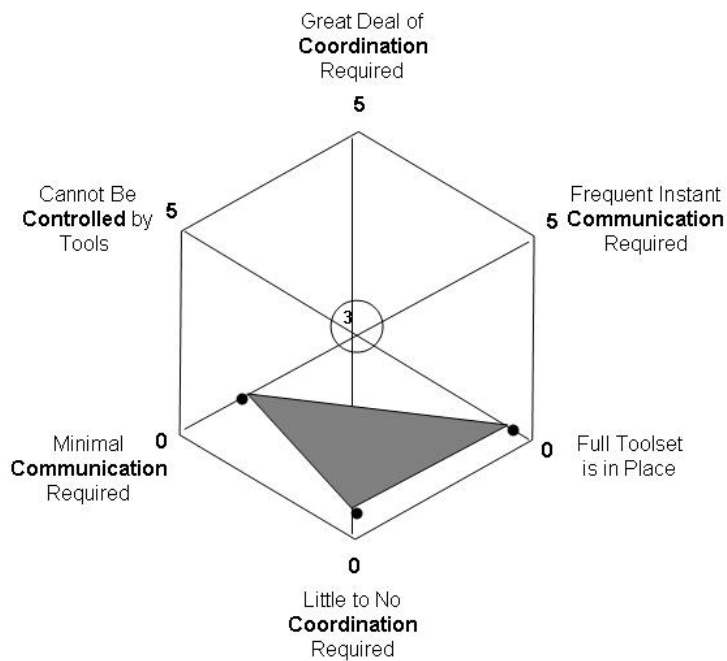
Examples are provided for your reference, below.



Example 1: Innovative project in an organization with high process maturity and sophisticated toolsets.



Example 2: Innovative project in a start-up company that has not yet completely defined its processes or purchased a complete toolset.




Example 3: Mature project that requires primarily maintenance/bug-fixing in a company that has mature processes and toolsets in place.


MATCHING YOUR PROJECT TO A VENDOR

With a visual description of your project now in hand, you can use the legend below to understand what outsourcing mix will best suit the project. This section of the paper provides a detailed discussion of each outsourcing option, and explains why some are better suited to certain types of projects than others. It also provides guidelines for how to evaluate different types of vendors.


LEGEND

 **Checkers = In-house or Near-shore**

If your project mapped mostly into the checkered area, it requires a high degree of coordination and communication, with very few tools or other controls in place to manage it. It may be a brand-new product or a very time-sensitive project. Such projects are best kept in house or outsourced to a partner in the same timezone and culture who is experienced working with early-version product development, and who is familiar with your target market.

 **Dots = Offshore**

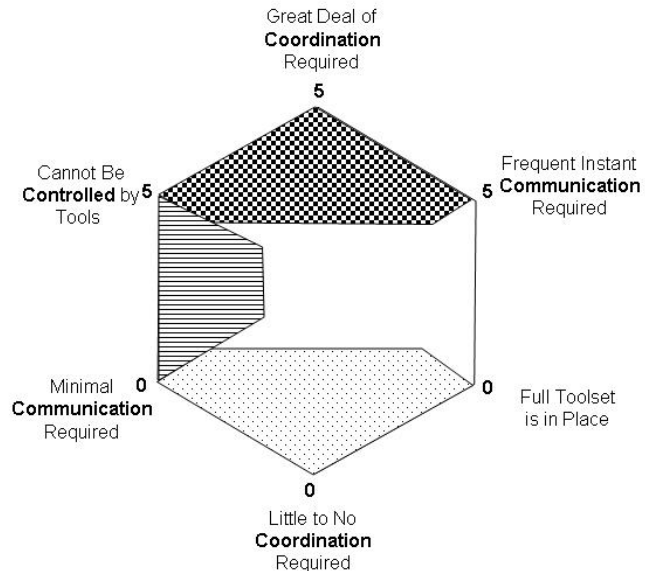
If your project maps primarily into the dotted area, it is likely a mature product that requires maintenance and bug fixing. You have strong controls in place, and both the code and the product have been documented comprehensively. These projects can be effectively outsourced to low-cost, offshore vendors.

 **White = Blended Model**

If the bulk of your project map falls into the white area, then it will be worthwhile for you to consider a blended outsourcing model. Blended models are the most flexible of outsourcing options, and they are discussed further on in this paper.

 **Lines = Time to Regroup**

If your project mapped into the lined zone, then it's time to step back and assess what's wrong. Either you plotted the map incorrectly, or something is fundamentally amiss with the project as it is conceived. As a rule, when coordination and communication increase, control decreases. With less control, greater human interaction is required.



In-house Considerations

The advantages of using an in-house development² team are obvious to any R&D manager. You have the greatest control over an in-house team and communication can be accomplished simply and quickly. Your in-house team likely includes some star players who are invaluable at particular types of innovation or technological problem solving. With an in-house team, logistical distractions such as language, culture and time zone are never issues (although human resources challenges do come into play). Focusing an internal team on development of core products also allows an organization to monitor security of its intellectual property (IP).

These advantages make in-house development desirable for core IP and for products that require a great deal of communication and coordination, as well as those lacking controls. Brand new product development or short-term, time-sensitive projects are examples – these require a level of innovation, experimentation and iteration that can be disruptive within an organization.

Still, it is often not feasible to assign highly iterative projects to in-house teams. Some considerations may include:

- **Resource availability** – internal resources may be fully engaged working on other high-priority projects.
- **Skills** – your in-house team may not have the required expertise for a particular project, especially if the project is an “experiment” or a one-off design.
- **Cost** – for one-off projects, such as customer-specific feature development, it is often too costly and distracting to move an internal team off of existing projects and ramp them up for a short-term project.

In situations like these, it is often a better choice to avoid the disruption by having an external team tackle the job. This may result in “nearshoring” or the use of local contractors.

Nearshore Considerations

Also called “onshoring,” nearshore outsourcing describes the practice of using software development firms that are “near” to you in timezone, culture, or location. In the best nearshore relationships an R&D organization reaps the advantages of outsourcing without the difficulties that offshoring can entail. Linguistic, cultural and time zone considerations are not an issue, and the R&D organization can use the onshore partner to accomplish projects that in-house teams may be unable to tackle due to availability, skills gaps or cost considerations detailed above.

Over time, a good nearshore software development partner can learn your market and your business inside out and add exceptional value. A skilled nearshore software development partner is typically well equipped for projects that entail a high degree of innovation, iteration and experimentation. Because an onshore partner will share your culture and ideally have a wide range of relevant product and industry experience, a great deal can go unsaid or undefined. The partner will be able to intuit your needs to a degree that is not achievable when working with offshore partners. In a strong onshore relationship, the partner will be able to “read between the lines and finish your sentences.”

² Note that in-house development is sometimes referred to as “inshoring”, but this use of the term can be misleading, because it is more often used to refer the practice of outsourcing to one’s own offshore captive development center.

This type of relationship can produce the highest quality results with the least amount of management overhead in the following situations:

- **Highly innovative projects.** For example, where only a “napkin sketch,” rudimentary design, or list of features/requirements exists.
- **Short, time-sensitive projects** that would be distracting for an in-house team to tackle, and which may derail core product development.
- **Non-core projects** requiring specialized skill sets. If the expertise required is not core to your R&D strategy, then it won’t make good business sense to temporarily hire the skills in-house. A nearshore partner can take on non-core software development with no disruption to your own R&D team. Examples of non-core projects may include porting existing products to new platforms, developing new or experimental features, and working on custom development projects for specific customers.
- **Below-1.0 development.** The advanced-concept phase of development is highly iterative and time sensitive, with a high degree of rapid interaction required between developers, project managers, and trial customers. Therefore, the development team needs to remain tightly integrated. A nearshore team in the same timezone and with appropriate experience is often as suitable for this type of project as an in-house team.

In such situations, cost is not the primary consideration. Getting to market first with an innovative, creative product is of paramount concern, and a nearshore partner with the right experience is ideal to accomplish that.

What to Look For

The best firms for innovative product development are located primarily in Canada, the United States and Israel. Firms in Eastern Europe are also emerging that show an aptitude for this type of development. When evaluating a nearshore partner, the following are the primary considerations:

- **Product development outsourcing (PDO) experience.** The vendor should be specialized in the development of products for the marketplace. This is in contrast to firms that focus on IT systems development and outsourcing.
- **Process and controls.** What processes and tools does the partner have in place to ensure that the project will be managed efficiently and that the final product will be of high quality? Are those processes agile enough to deal with ongoing change? In addition to a documented process and toolsets for version control, bug tracking, etc., look for strong experience and references on short-term and highly innovative projects.
- **Market and industry knowledge.** How familiar is the partner with your market and industry? Will they be able to ramp-up quickly and understand preliminary concepts sufficiently enough to flesh them out independently?
- **Technology expertise.** Does the partner have the right technical skills for the project? Beyond this, you will want a partner with deep technical skill that will enable them to “look under the covers” and develop what you need – not just what you ask for.
- **User experience design capabilities.** A resident UI design team or expert is a good indication that the vendor has this. UI design skill is critical for the introduction of any product to market, and high quality design prototypes also provide an excellent means of communicating intent.

- **A stable team with low turnover.** Attrition of key team members can cause turmoil and set a time-sensitive project back dangerously. Look for proof that the company has been stable over the long term and through economic fluctuations. Their business should be diversified over at least a few industries and the majority of their revenues should not come from a single client.
- **Track record of innovation.** If, in addition to stability and process controls, the vendor can demonstrate deep experience in highly innovative and iterative projects, then you'll know that their processes are agile.

Offshore Considerations

When there are no sentences left to finish and the product is well documented and easily understood, then cost should be the primary consideration. In product maintenance scenarios it's critical to keep margins low, and that's where offshore software development partners are valuable. Offshore partners are also often the right choice for the development of new products that have detailed design specifications.

If one or more of these criteria are in place, then an offshore partner is likely a good choice for your project:

- The product is at **version 6.0 or higher**, and needs to be maintained through a structured bug-fixing process.
- Comprehensive code and product **documentation** exist.
- A very **detailed design specification** and a detailed UI specification exist, fully fleshed out.

What to Look For

Typically located in India, Russia, South America, or East Asia, the best of these firms will demonstrate mature processes and a structured environment that minimizes surprises and keeps costs low. Industry and product-specific experience are not of primary importance when evaluating an offshore partner. Instead, look for:

- **Controls.** Are the vendor's tools and controls comprehensive? At least 80% of the offshore vendor's process should be automated.
- **A good project manager** with some understanding of your culture and market, and excellent skills in your language. This person will become your "lifeline" to the project.
- **A planned review cycle**, which the vendor should create in concert with you.
- **Transparent tools** that give you full visibility into metrics such as bugs reported versus bugs fixed, mean time to repair, and so on. The tools should work well in a distributed environment.
- **CMM level 3** or higher
- **Low cost**
- **Good technical skills**
- **Financial stability** and a mature business

Blending Considerations

A blended outsourcing model is the most mature of outsourcing strategies, and the vast majority of the world's largest R&D organizations use this type of model. In a blended model, a variety of outsourcing options are used at different points throughout the organization and throughout the lifecycle of a single product. This is a simple example:

- In-house development team focused on the development of core IP

- Local contractors used to supplement in-house resources on advanced-concept development, and for quality assurance
- Onshore partner(s) used for new feature development, platform porting, and customer-specific customizations
- A captive offshore partner (“insourcing”) for the development of clearly defined new products or product iterations
- Mature products transitioned to offshore partner(s)

It is useful to keep in mind that some onshore partners themselves use a blended outsourcing or dual-shore strategy. For example, the vendor may blend offshore resources from regions like Eastern Europe, who work directly with the vendor’s onshore team and project manager, to attain an appropriate mix of speed, quality, skill and cost for the project. Or, once a new product design is frozen, the vendor may outsource some of the development to an offshore alliance. This provides the R&D organization with rapid, high-quality innovation while helping to minimize the costs of new product development.

When using a blend of outsourcing resources on a single project, it is imperative to have a project manager who is close to you in time, culture and experience. No matter what mix of outsourcing the project entails, you’ll need a single point of contact who maintains a grasp of the status of all aspects of the project’s progress, and who can realign resources to manage slippage.

THE REAL 3CS: COMMUNICATE, COMMUNICATE, COMMUNICATE

One of the most critical aspects of any software development project’s success is communication – that requirement doesn’t change when the project is outsourced. It is especially important in a blended model, where a number of contributors and simultaneous efforts exist.

No matter what type of project or outsourcing strategy you embark upon, be prepared and willing to communicate your expectations for communication. Do you want to be involved at every stage or only particular milestones? What type of updates and project reports will you consider useful, and how do you want delays, changes, and deliverables to be conveyed to you? One of the most effective ways to ensure that a project’s communication will be successful is to look for partners and project managers who listen, ask good questions, and demonstrate an understanding of your needs out of the gate.

