Software development practices
White paper
November 2008

Rational® software

Enable the agile enterprise through incremental adoption of practices.

Per Kroll, chief architect, expertise development and innovation, Rational software, IBM Software Group

Contents

Improve IT results by leveraging lean and agile principles

IT organizations are struggling to do more with less. And they're not getting a lot of encouragement. Less than 40 percent of stakeholders are satisfied with the speed of internal application development, and not even half of them are satisfied with the quality of those applications. There is no silver bullet to fix these problems; however, we can learn from other industries how to improve quality, customer satisfaction and productivity.

Lean software development has its roots in Toyota's lean manufacturing techniques, and according to many, it provides an intellectual foundation for agile development. Some of the underlying thoughts behind lean and agile development include:

- ***Build software through teamwork.*** *If you want to get the most out of your team members, you need to give them the responsibilities and tools that help them innovate and contribute to the success of a project. This includes ensuring that they have the underlying infrastructure that allows them to collaborate across cultures, time zones and geographical boundaries.*
- ***Minimize waste and shorten the value chain.*** *Streamline development by removing intermediate stockpiles of noncode artifacts that grow old. Instead, you need just enough documentation and a process to enable your organization to rapidly translate initial investments in requirements into validated code that adds value to the business.*
- ***Put quality first.*** *Many assume that high-quality code has to cost more. The fact is, it's often more expensive to build low-quality code. Why? Traditionally, you dedicate testers to test code and find defects that developers need to understand and correct. Testers then reverify that the problems have been addressed. Having developers instead test their own code through developer test practices, such as test-driven development, and find defects early in the process, through automated regression testing, can help you realize a very high ROI.*

Highlights

- ***Provide rapid feedback and response.*** *The process of developing software is filled with uncertainties, and agility provides an effective approach for dealing with uncertainty. Unclear what the right solution is? Use iterative development with continuous feedback and periodic course corrections. Don't know which architecture to use? Implement and test key aspects of the architecture early on to gain feedback on what works. Unsure of whether the many components being developed can work together? Apply continuous integration and testing. All of these are agile approaches.*

The business organization and the IT organization share the same goal: to maximize business value by leveraging available resources.

- ***Promote one extended team.*** *Many organizations perpetuate an "us versus them" mentality between the business organization and the IT organization. However, both share a common goal: to maximize the value for the business by effectively leveraging all available resources. For this to be a joint goal, all departments need to be equally vested in success and collaborate as one team.*

Being agile in challenging environments: agility at scale

Given that the principles of lean and agile development apply to a broad set of industries and a broad set of contexts, the concrete practices required to implement these principles vary greatly. Agile development is often described in terms of practices that work well for small, colocated teams developing applications for internal use. In this context, the hallmarks of agile development are quickly recognized: daily stand-up meetings, requirements and tasks managed via whiteboard, constant access to customers and one-room team gatherings. These are all good things to strive for, but you may work in contexts where these practices need to be augmented or changed to deal with such challenges as:

Many organizations know the value of agile development, but their work contexts are such that agile practices need to be scaled up or changed to be effective.

- *Application complexity.*
- *Geographical distribution.*
- *Compliance requirements (such as Software Engineering Institute Capability Maturity Model Integration [SEI CMMI], Control Objectives for Information and related Technology [COBIT], Sarbanes-Oxley Act [SOX] and Basel II).*

- *Enterprise disciplines such as enterprise architecture or waterfall-like program management processes.*
- *Organizational distribution such as outsourcing relationships or partnerships.*
- *Degree of governance required, including formal application lifecycle processes with milestone sign-offs by a broad set of stakeholders.*
- *Large team sizes.*

With the increased importance of IT, which continues to evolve at ever more rapid rates, it is important to document how teams must collaborate to effectively develop software. To effectively provide support for agility at scale, the following is required from your process:

- *It needs to have as* ***little ceremony*** *as feasible for your context.*
- *It needs to be* ***adaptable*** *and address the many contexts your team may face.*
- *It needs to allow the team to take* ***ownership*** *of the process that works for its context.*
- *It needs to* ***scale*** *to address the complexity factors that the team is facing.*
- *It needs to provide an effective* ***learning solution*** *to ensure that the team understands or can learn what drives success.*
- *It needs to be* ***measurable*** *so you can understand what works and what doesn't and take corrective actions when necessary.*

Highlights

For agility practices to work, your development process should be simple, adaptable, scalable, educational, measurable and owned by the team.

Highlights

When you build an effective process from a collection of practices, you can customize the solution and incrementally evolve it to meet your needs.

A practices approach to adopting a process can help you address practitioner pain points and ensure that the evolving process is satisfying business objectives.

Practices: changing the way processes are communicated

Building a tailored process from a collection of practices (see figure 1) provides you with the flexibility to rapidly evolve the process to address your business needs. Examples of practices include iterative development, continuous integration, evolutionary architecture, shared vision and test management. Practices not only provide guidance and techniques within a certain domain; they come with an underlying philosophy of how to do a certain aspect of development a certain way. For example, if you do not like the philosophy behind evolutionary architecture, where the architecture is grown over time, you can replace it with another practice, such as doing a big, up-front architecture or doing no architecture at all. Here are some of the advantages of a practices approach to adopting process:

- ***Practices represent a useful unit of knowledge.***
- ***Practices can be independently and incrementally adopted.***
- ***Practices published for ease of use (for example, on an internal Web site) can provide one-stop access*** *to courses, tool features, services, articles, process content and enactment strategies. Hence, they can be helpful portals to information that enables effective implementation.*
- ***Practices map to practitioner pain points.*** *You can adopt a practice for just continuous integration, or you can adopt a comprehensive process.*
- ***Practices are building blocks for processes.*** *You can select a collection of practices that cover the full lifecycle. A practice can be a part of many processes, allowing content to easily be shared across a multitude of processes.*
- ***Practices map to organizational business objectives.*** *If your organization is trying, for example, to improve productivity, you can list a specific set of practices that can help you achieve that objective.*
- ***Practice adoption can be measured.*** *This allows you to track which practices you are doing well, and which you need to work on.*

Highlights

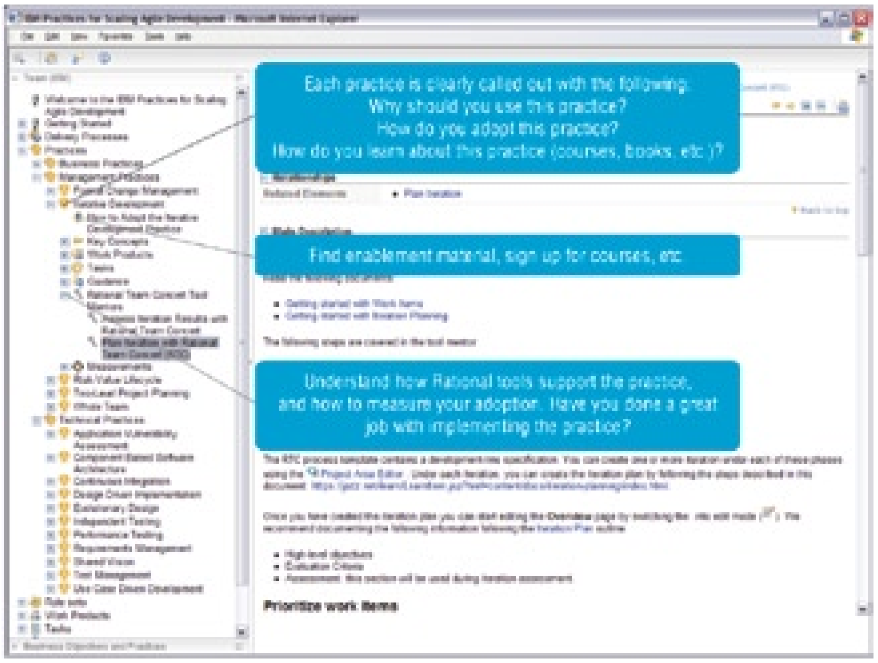
Each practice is clearly called out with the following:
Why should you use this practice?
How do you adopt this practice?
How do you learn about this practice (courses, books, etc.)?
Find enablement material, sign up for courses, etc.
Understand how Rational tools support the practice, and how to measure your adoption. Have you done a great job with implementing the practice?

Figure 1: A practice provides guidance on why you need it and how to get started with it, including reference and enablement material. It also provides guidance on tasks, artifacts and tools that help you implement the practice, and it provides measures to track the level of adoption.

How IBM Rational Method Composer leverages practices

To help you get started with a practices approach to process, Rational Method Composer features a set of practices that can be further extended with your own practices.

IBM Rational® Method Composer software features a flexible process management platform with a comprehensive process library. Offering content on a broad variety of topics, the process library provides information on software and systems delivery (from business modeling and requirements management to configuration management and testing), on service-oriented architectures (SOAs) and SOA governance, on asset-based development and asset governance, on operations and systems management; and on many other subjects.

Rational Method Composer also allows you to author your own practices by using a tool to manage, reuse, configure, tailor, publish, report on and deploy content. While enabling you to write your own processes from scratch, Rational Method Composer also lets you use its content as a starting point, create derivatives to reflect your style of development, and integrate your

Highlights

techniques with industry best practices. As IBM produces new versions of the process library, you'll be able to transfer your process assets to the new version and easily maintain your differentiating process assets.

Rational Method Composer also now supports the notion of independently authored and consumable practices. Even though they can be used for agile or non-agile development, Rational Method Composer practices are currently optimized for agile teams, allowing them to address many of the complexity factors described earlier. The practices are logically divided into the agile core and a set of extensions, as shown in figure 2.

Rational Method Composer practices can be used for agile or non-agile development, but they are logically arranged with an agile core and a set of extensions.

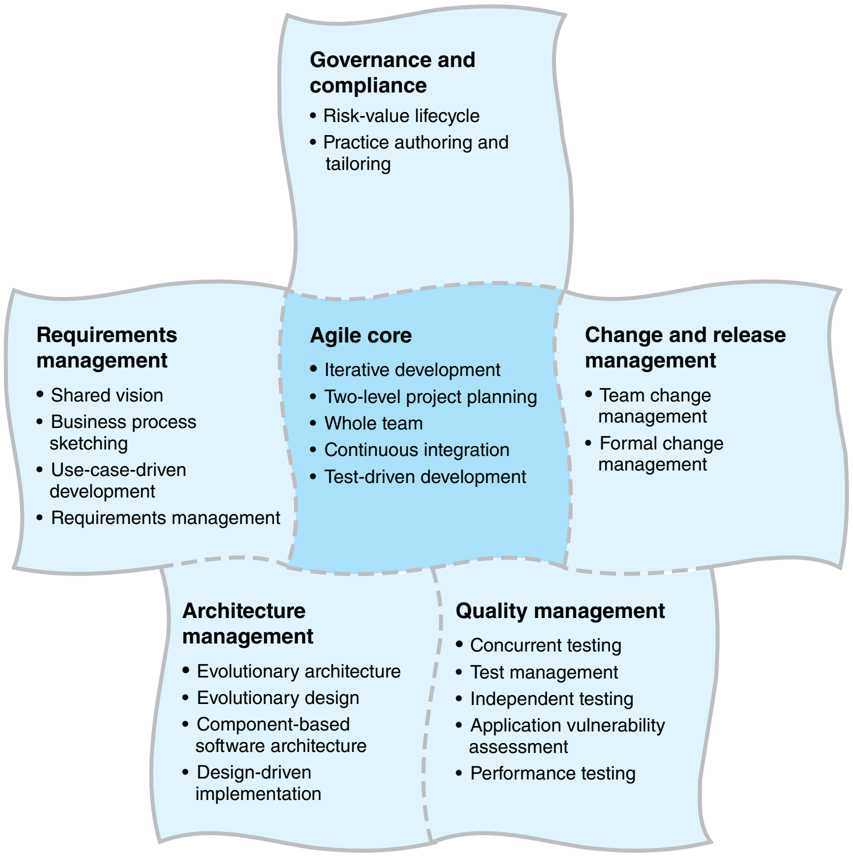
Governance and compliance
• Risk-value lifecycle
• Practice authoring and tailoring
Requirements management
• Shared vision
• Business process sketching
• Use-case-driven development
• Requirements management
Agile core
• Iterative development
• Two-level project planning
• Whole team
• Continuous integration
• Test-driven development
Change and release management
• Team change management
• Formal change management
Architecture management
• Evolutionary architecture
• Evolutionary design
• Component-based software architecture
• Design-driven implementation
Quality management
• Concurrent testing
• Test management
• Independent testing
• Application vulnerability assessment
• Performance testing

Figure 2: IBM Rational practices are divided into an agile core with practices recommended for teams striving to become agile. The practices also include a set of extensions to be considered based on the complexity factors you face.

Highlights

The agile core contains five practices that should be on your shortlist for implementation: iterative development, two levels of planning, whole team, continuous integration and test-driven development.

The agile core

The practices in the agile core are recommended for any team striving to adopt agile methods. This does not mean you cannot become agile without them, or that you have to adopt all of them; it simply means that these are the practices that should be on your shortlist for consideration:

- ***Iterative development.*** *Deliver working software every few weeks. Get feedback from end users to ensure that what you build delivers maximum business value.*
- ***Two levels of planning.*** *Conduct coarse-grained project planning up front—often referred to as release planning—and then do detailed planning for the current and next iterations. Revise your project plan as needed based on actual iteration results.*
- ***Whole team.*** *Ensure close collaboration and transparency among motivated team members as a means to minimize intermediate artifacts and associated waste. Self-organize around component teams responsible for development, building and testing.*
- ***Continuous integration.*** *Produce working builds that are automatically and frequently tested to ensure continuous feedback.*
- ***Test-driven development.*** *Developers test their own code to avoid introducing and removing defects, reducing the time needed to integrate and stabilize builds.*

These practices provide a core foundation for agile projects, and practices can be added to address the specific needs of the team. Some teams may not yet be ready or able to adopt all the core practices, and may, for example, deploy a subset of the core practices with a few of those practices that extend the core.

Highlights

Extending the agile core

The agile core provides a good starting point for teams, but the core practices alone are often insufficient to address the needs of most teams. Teams should consider the practices that extend the agile core and that provide specific benefits. IBM, IBM Business Partners and the broader software and systems delivery ecosystem—including the Eclipse Process Framework open source community[1]—all contribute to the practices, which will likely grow over time.

Here is a sample of the current practices that extend from the agile core:

- ***Shared vision.*** *The team, including all stakeholders, is united around a common vision, providing guidance on what the end results should look like.*
- ***Use-case-driven development.*** *Usage scenarios become a virtual collaboration space for a cross-functional team of users, analysts, developers and testers to ensure that the functionality that was developed is the functionality that was needed.*
- ***Evolutionary architecture.*** *The architecture is grown incrementally, and it is continually validated to ensure that the application is easy to maintain and able to address nonfunctional requirements.*
- ***Component software architecture.*** *While this practice applies the same underlying philosophy of incrementally growing and validating the architecture, it puts a stronger focus on early stabilization and more formal documentation of the architecture. This practice helps address the needs of complex systems with large and distributed development teams.*
- ***Concurrent testing.*** *This practice adopts testing throughout an iteration, concurrent with development. This prevents teams from compressing testing into a separate activity at the end of an iteration or release.*

By themselves, practices in the agile core are often insufficient for teams. Consider adopting practices that extend from the core and are supported by the broader software and systems delivery ecosystem.

Highlights

- ***Independent testing.*** *This practice promotes concurrent testing, but it also enhances the efforts of more mature test organizations—typically those that develop mission- or safety-critical systems—that are looking for a reliable, reproducible way of testing their products.*
- ***Application vulnerability assessment.*** *This practice describes how security groups and development teams can effectively collaborate during the software development lifecycle to assess the vulnerability of Web applications.*

The beauty of a practices approach to process is its ability to incrementally and flexibly evolve to reflect different development styles and to meet changing needs.

Many teams may not need all of these practices. For example, some teams may realize that, after a while, the concurrent testing practice is no longer rigorous enough for a growing project that has increasing demands on reliable testing. In this case, the team can switch to the independent testing practice. This is the beauty of practices: You avoid painting yourself into a corner, and instead are allowed to evolve your process for the task at hand and for the culture of your particular team. Team members can also update the practices to reflect their style of development and lessons learned, which allows their practices to be harvested and leveraged throughout the organization.

Building your process from practices

IBM Rational Method Composer makes it easy to choose the practices you need. It provides a process builder that uses a shopping cart metaphor to walk you through a set of selection criteria. You can browse practices before selecting one, and you can see what the resulting process looks like as practices are added. When you're finished, you can choose to publish the process as HTML to your desktop or Web server; you can export it to XML; or you can export it as a work-breakdown structure to a project management tool.

Highlights

Rational Method Composer allows you to build your process by browsing scenarios and selecting criteria that you'd like your end process to meet.

Using the process builder, you can build your own selection criteria using out-of-the-box samples. One selection criterion walks you through the software development lifecycle, asking you what practices you want to select for management, requirements, architecture management, development and so on. For each topic, a set of plug-ins is listed. Another sample selection criterion walks you through common operational objectives, such as improve quality, reduce time to value and increase innovation. For each objective, practices that help you achieve the goal are listed. You can also create your own decision criteria using, for example, SEI CMMI process areas or other categorizations. It is easy to align available practices with your desired categorization schema.

Addressing Scrum and other processes through practices

Practices are largely process neutral. A practices approach moves the discussion from "Is my process better than your process?" to "Which practices are the most relevant for this context?" This means you can assemble a process by selecting a set of practices and combining them into a lifecycle defined by your process.

For example, the practices iterative development, whole team and two-level planning can be combined with Scrum's roles to arrive at a process very similar to Scrum. Or you can take the test-driven development, continuous integration and rapid testing practices, add the XP roles and get a process very similar to XP. If you combine all of those practices with the shared vision, use-case-driven development, risk-value lifecycle, evolutionary architecture, evolutionary design and team change management practices, and then add Open Unified Process (OpenUP) roles, you'd get OpenUP. Add yet more practices, and you can re-create variants of the IBM Rational Unified Process® methodology—see figure 4.[2]

A practice-based approach transforms the question of whether one methodology is better than another into a discussion about which practices are better for the business.

This practice-based approach transforms the question of whether one methodology is better than another to a practical basis for agreement: "We need these three practices. Which of these nine other practices do we also need?" This is a much more productive discussion.

Highlights

Rational Method Composer allows you to map CMMI process areas to the practices that address them.

Addressing CMMI through practices

Many organizations leverage assessment frameworks such as SEI CMMI to leverage the benefits associated with being certified at a certain level, or to qualify to do business with certain entities, such as the United States Department of Defense. However, CMMI and similar assessment frameworks do not guide teams in how to develop software; they only define maturity levels and give requirements regarding the process that must be followed. Rational Method Composer and practices can help. For each of CMMI's process areas, you can map to one or more practices that address it. This means you can easily produce a minimal process that addresses just the areas you are interested in. Then you can easily evolve your process to address additional areas in the future.

CMMI Level 4 focuses on measuring the effectiveness of your process so you can take corrective actions, and Level 5 focuses on ensuring that process improvements are driven from your business objectives. These thoughts are foundational elements of the Measured Capability Improvement Framework,[3] as described in the next section. But why should only CMMI Level 4 and Level 5 organizations benefit from such foundational principles? The Measured Capability Improvement Framework is designed to measure and steer improvements based on an organization's most important business objectives, which means it's applicable to organizations at any CMMI level.

Highlights

By combining different sets of practices and applying different roles, you can construct processes that are similar to RUP, OpenUP, Scrum and XP.

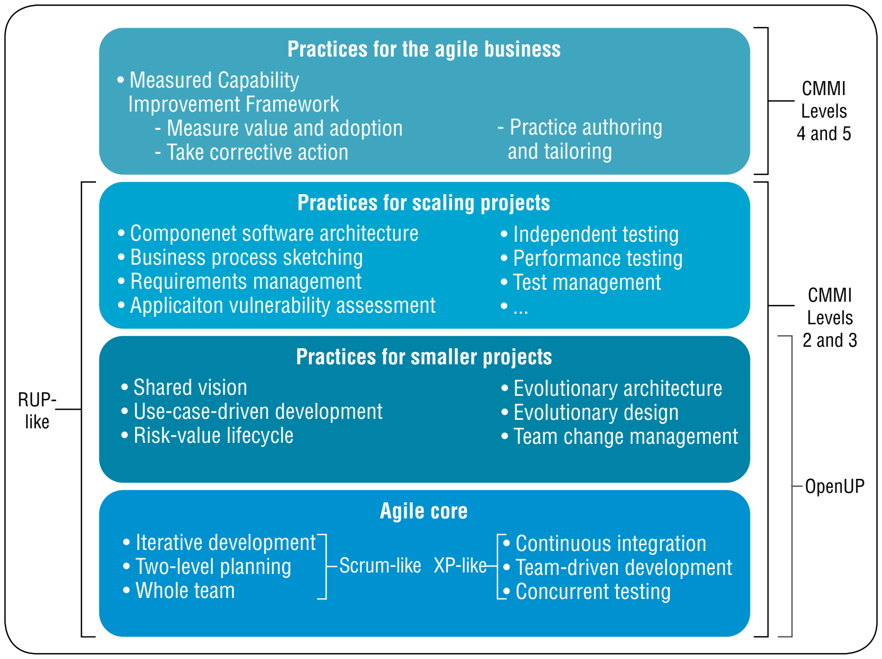
Practices for the agile business
• Measured Capability Improvement Framework
- Measure value and adoption
- Take corrective action
- Practice authoring and tailoring
Practices for scaling projects
• Componenet software architecture
• Business process sketching
• Requirements management
• Applicaiton vulnerability assessment
• Independent testing
• Performance testing
• Test management
• ...
Practices for smaller projects
• Shared vision
• Use-case-driven development
• Risk-value lifecycle
• Evolutionary architecture
• Evolutionary design
• Team change management
Agile core
• Iterative development
• Two-level planning
• Whole team
Scrum-like
XP-like
• Continuous integration
• Team-driven development
• Concurrent testing
RUP-like
CMMI Levels 4 and 5
CMMI Levels 2 and 3
OpenUP

Figure 3: By combining different sets of practices and applying the right roles, you can construct a variety of processes. You can also map practices to different process areas within SEI CMMI, allowing you to create minimal processes that address only the areas you are interested in.

A systematic and measurable approach to agile adoptions

We have discussed the benefits of agile, the challenges with agility at scale, and how IBM Rational Method Composer can help you to adapt your processes through the notion of consumable and incrementally adoptable practices. But how do you succeed in transforming your organization to an agile mind-set? The Measured Capability Improvement Framework approach can help.

Highlights

The Measured Capability Improvement Framework can help you improve software and systems delivery by executing on the four phases for process improvement: identify desired business objectives, produce a roadmap to delivery excellence, execute on the roadmap, and measure and adopt.

Leveraging IBM's years of experience driving software and systems delivery excellence, the Measured Capability Improvement Framework from IBM provides a systematic approach to incrementally improving your software and systems delivery capability. The framework accomplishes this by putting into practice the four phases for well-governed process improvement:

- ***Identify desired business objectives.*** *The Measured Capability Improvement Framework begins with a review of each customer's unique challenges and business goals. These starting points are then mapped to standardized operational objectives such as improve productivity, reduce time to value, improve quality and increase innovation—helping to ensure that improvement efforts focus on the initiative that will bring the biggest value.*
- ***Produce a roadmap to delivery excellence.*** *IBM works closely with customers to identify and select target practices, products and services that will drive desired business objectives. Using assessments and out-of-the-box mappings of business objectives to practices, IBM develops a roadmap based on your current strengths and weaknesses and your desired future state.*
- ***Execute on the roadmap.*** *To effectively deploy well-governed practices and associated products, organizations can use a variety of assets that help lower the overall cost of deployment. Deployed practices become a part of the development environment, helping to build team cohesion and an understanding of the delivery approach required to achieve expected business results.*
- ***Measure and adapt.*** *To measure the results of an improvement effort, organizations can use metrics at three levels. Measurements at the practice level provide an understanding of the inner workings of the software and systems delivery organization and of how well practices and associated products have been adopted. Measurements at the business and operational levels provide an understanding of whether desired business outcomes are achieved. Using these metrics, organizations can make adjustments as needed.*

Highlights

Large or small, agile or not, your team can work together more effectively and improve software and systems delivery by incrementally adopting practices that address various business and operational needs.

Conclusion

Lean and agile development allows IT organizations to react to current challenges. If your team strives to be agile, Rational Method Composer offers practices that can help you achieve that goal. Providing a rich library of practices that can be assembled into processes, Rational Method Composer can help you bring your team together to work effectively. Processes can be adapted and adopted incrementally as projects progress and business needs change. And if your team tackles challenges such as complex applications, geographical distribution, large team sizes or compliance needs, Rational Method Composer offers practices to help you deal with those issues. Piecemeal or in whole, Rational Method Composer practices can be deployed in a way that allows them to become a natural part of your development environment.

To support the introduction of new practices into your environment, the Measured Capability Improvement Framework can help you select and incrementally adopt the practices that have the biggest business value for your specific context. The framework can also measure the results of that adoption to see where modifications can be made to ensure that expected business value is achieved.

For more information

To learn more about IBM Rational Method Composer software, IBM agile solutions and the Measured Capability Improvement Framework, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/rational/rmc

ibm.com/rational/agile

and

ibm.com/software/rational/announce/mcif

1 See www.eclipse.org/epf.

2 IBM intends to add practices to later versions of the IBM Rational Method Composer product, thereby enabling users to re-create most variants of the RUP solution.

3 Please see the recent IBM white paper *The Measured Capability Improvement Framework: a systematic approach to software delivery excellence* for more information.

RAW14077-USEN-00