Developer Tools and Services Roadmap

v0.7

8 July 2008

Developer Tools and Services

- MIT Application Platform:
  - Software stacks: JEE (SASH), PHP, MySQL with associated toolkits
  - APIs, documentation, reference implementations to access MIT infrastructure services
- Developer Tools:
  - Tools used by developers to create and manage code and processes
- Developer community and support
  - MAP Working Group, Steering Committee

Vision

- Provide software stacks, APIs, toolkits, documentation and developer tools in order to:
  - Lower the cost of SW development at MIT
  - Produce quality software
  - Rapidly develop SW in response to changing needs
  - Improve consistency and predictability
- Foster a developer community that is actively sharing tools, reusable code, and best practices

Goals

- Developers can build applications from a toolkit of modular, flexible parts, rather than build all the components themselves every time
- Developers can integrate with MIT's infrastructure services through appropriate interfaces
- SW projects have state of the art tools to facilitate best practices
- Social computing & organizational infrastructure is in place to foster developer community across MIT
- MAP Working Group, Steering Committee are actively setting priorities and guiding development

Value to the Community

- Consistency of development practices and tools improves predictability of projects
- Re-use of code and components improves efficiency of development cycle
- More uniform user experience
- Standard tools and practices make it easier to integrate developers, 3rd party packages
- MAP is community-driven, and therefore should meet its needs

Trends/Drivers

- MIT's software infrastructure needs to continue to evolve to meet the needs of current IT practice
- We need to position ourselves to deliver software every day
- Student VISION will have a big impact on MIT
- Service-oriented Architecture
- Rich Internet Apps (Web 2.0)

Current State: Stacks and Toolkits

- Assets:
  - SASH stack for Java
  - Working on a stack for PHP
  - JQuery for AJAX
  - SOAP services with WSDL
- Gaps:
  - incomplete library of reference implementations, documentation for integration with our infrastructure
  - Many developers unaware of supported options, just use the quickest and easiest thing
  - DLCs have small, one-person projects using the latest technology, not necessarily looking for mature, integrated, supported stacks

Current State: Dev Tools

- Assets:
  - Source control (SVN)
  - Build Dependency management (Maven) underway
  - Continuous Integration (Bamboo) underway
  - IDE (MyEclipse) – site license available
  - Code browser (OpenGrok for Kerberos team)
  - Issue management (Jira)
- Gaps:
  - Code analysis
  - QA tools: Load and stress, automated functional testing
  - Integrated development infrastructure
  - No supported developer tools for the whole community

Current State: Dev Community

- Assets:
  - We have MAP Working Group, Steering Committee structure
  - SAIS is our biggest customer
- Gaps:
  - No developer portal yet (in development)
  - Non IS&T developers not engaged yet
  - A relatively low priority, all developers including ISDA developers have other jobs as well

End State: Stacks & Toolkits, 1

- Multiple stacks available to the community
- MAP working group, steering committee help define priorities for new stacks in response to community needs
- Stacks used by Student Vision, other IS&T development projects
- Small DLC projects use “the stacks” vs. non-scalable, one-off solutions

End State: Stacks and Toolkits, 2

- Minimally SOAP and REST APIs to MIT infrastructure from various stacks
- Complete set of reference implementations, libraries, doc to access MIT infrastructure from supported stacks
- Kuali-certified implementations as required for Student VISION

End State: Dev Tools

- Many MIT teams working in a comparable way re: source code management, continuous integration, best practices, testing
- Standards based so new developers, consultants able to come up to speed more quickly, less unique learning required

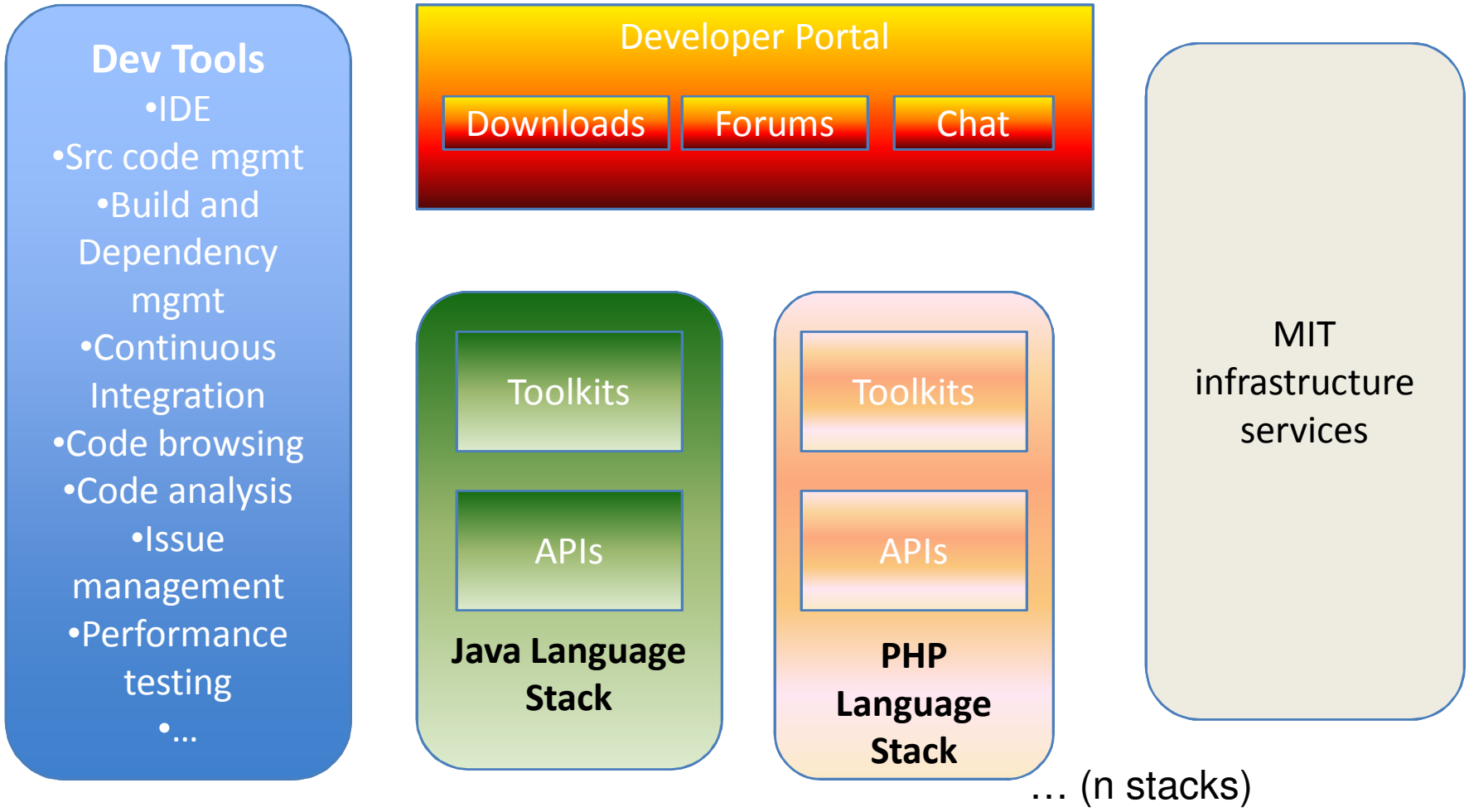
End State: Dev Community

- Active Developer Portal, where developers can exchange code, ask/answer questions, doc, etc
- ISDA, IS&T, and DLC developers are empowered to participate
- ISDA/IS&T provides sponsorship, invests in “Developer Support” (trained developers who will answer/help other developers)

Approach to Execution

- Stacks and toolkits:
  - Prefer open standards with strong support options
  - Integrate infrastructure services with supported stacks, provide doc, reference implementations, toolkits as needed
  - Take Kuali into consideration as it unfolds
- Dev Tools:
  - Use best of breed dev tools
  - Use them ourselves, make others want them
  - Research, prototype, test, use
- Community:
  - Build a dev portal, communication infrastructure
  - Sponsor MAP contributors to incentivize organizations

Conceptual Architecture



Characteristics of the Community

- Working Group are developers on the ground, Steering Committee have the long view
- Developers feel heard, not dictated to; there is perceived benefit to their daily work in using “the stacks” and tools
- Consensus is reached by focusing on practical value & core infrastructure, staying away from ideological purity, allowing choice
- ISDA/IS&T offers sponsorship, other groups buy-in to letting their developers contribute
- An iterative approach to keep up with evolving standards

Risks

- Failure to engage community means work will not be used: dev projects may re-invent the wheel, lessons not learned across the org
- Failure to attain management sponsorship jeopardizes developers ability to contribute, support their work for others
- Developer support can become a sink for ISDA

Benefits restatement

Who doesn't want

- Better
- Cheaper
- Faster?