MAP Roadmap

v0.6
2 July 2008

MAP

- MIT Application Platform:
  - Software stacks: JEE (SASH), PHP (Zend), 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

Vision

- Provide software stacks, APIs, toolkits, documentation and developer tools in order to:
  - ① Lower the cost of SW development at MIT
  - ② Produce better quality software
  - ③ Rapidly develop SW in response to changing needs
  - ④ Improve consistency and predictability
- Foster a developer community that is actively sharing tools, reusable code, and best practices

Goals

- Developers can build new applications from a toolkit of parts, rather than build all the components themselves every time
- Developers can integrate with IS&T’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
- Re-use of code and components improves efficiency of development cycle
- More uniform user experience
- Easier to integrate new developers, new 3rd party packages
- MAP is community-driven, and therefore should meets its needs

Trends/Drivers

- MIT's software infrastructure was very advanced 15 years ago – now it needs updating
- Big SW projects that take forever to deliver are the past – needs and technology change too fast
- Student VISION will have a big impact on IS&T
- Service-oriented Architecture
- Rich Internet Apps
- Small applications proliferate in DLCs across MIT, represent possible liabilities (security, confidentiality)

Current State: Stacks

- Assets:
  - SASH stack for Java
  - Working on a Zend/Drupal stack for PHP
  - JQuery for AJAX
  - MySQL cluster underway
- Gaps:
  - Many developers unaware of supported options, just use the quickest and easiest thing
  - DLCs have small, one-person projects using the latest technology, at odds with more conservative supported stacks

Current State: APIs

- Assets:
  - SOAP services with WSDL
- Gaps:
  - incomplete library of reference implementations and documentation for integration with our infrastructure

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
  - Load and stress testing tools
  - Automated functional testing tools

End State: Stacks

- Stacks for Java, PHP
- Shared MySQL cluster
- 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 one-person DLC projects use “the stacks” vs. non-scalable, one-off solutions

End State: APIs

- SOAP and REST APIs to MIT infrastructure services
- Complete set of reference implementations and libraries to access MIT infrastructure from Java and PHP
- Kuali-compatible implementations as required for Student Vision

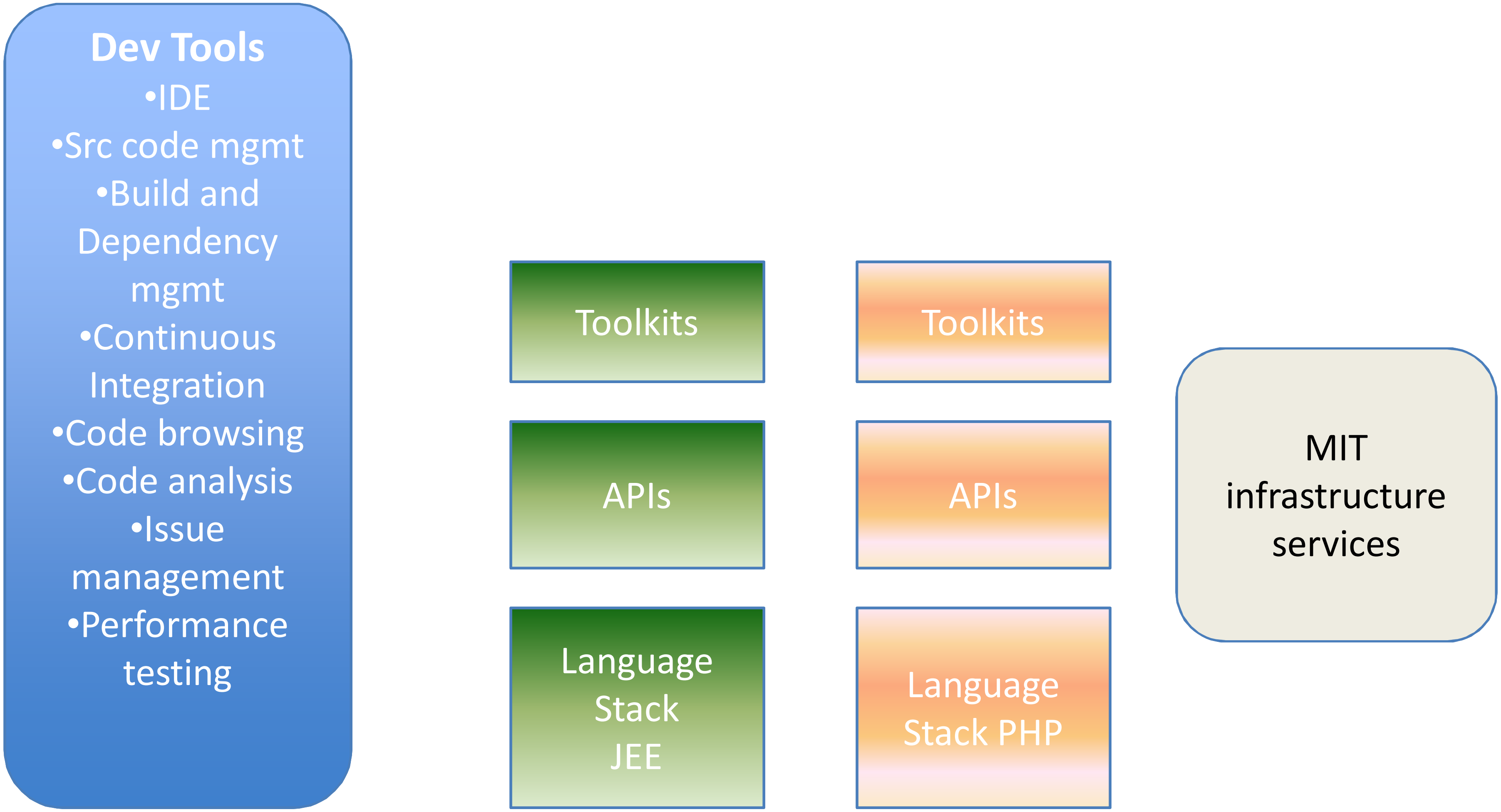
End State: Dev Tools

- 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

Approach to Execution

- Stacks:
  - Prefer open source, widely accepted standards with paid support when possible
- APIs:
  - Top priority: Integrate identity services with JEE, PHP stacks
  - Take Kuali into consideration as it unfolds
- Tools:
  - Use best of breed dev tools; open source when possible, but commercial is acceptable
  - Use them ourselves, make others want them
  - Research, prototype, test, then turn over operation to OIS
- MAP working group, steering committee help set on-going priorities as work develops
  - Need to engage "non-committed developers" in DLCs as well
  - Need to "sponsor" MAP contributors to incentive organizations

Conceptual Architecture



Dev Tools

- IDE
- Src code mgmt
- Build and Dependency mgmt
- Continuous Integration
- Code browsing
- Code analysis
- Issue management
- Performance testing

Toolkits

APIs

Language Stack JEE

Toolkits

APIs

Language Stack PHP

MIT infrastructure services

Dependencies/Assumptions

- JEE remains the major development stack, growing use of PHP
- Kuali will be driving Student Vision, and Student Vision will be driving a lot of new software development
- SAIS is our biggest customer, but isolated developers in DLCs are very important to engage
- Tools will keep evolving, we will never be “done”
- MAP Working Group, Steering Committee is our governance structure
- Standard tools & practices make it easier to ramp up developers, work with 3rd party packages & projects

Risks of Not Doing

- Individual development projects cost more, take longer, re-invent the wheel over and over
- Standards are not adopted, little re-use or compatibility
- Liability if rogue DLC developer compromises confidentiality thru local apps
- Harder to find 3rd party developers who have the skills

Risks of Doing

- It takes longer to build infrastructure, vs. just “go do it” on projects
- Wasted effort because developers don't use it
- Partners in the community must build these tools and make sure they work, will other bosses provide the resources?
- Developers like to argue about tools and techniques; not always easy to get agreement
- Standards are always evolving, need to develop an iterative approach to providing these tools
- Need a “de-support” strategy as tools evolve

Benefits restatement

Who doesn’t want

- Better
- Cheaper
- Faster?