Distributed Planning Poker

Integrating IBM Rational Team Concert and Google Wave for distributed effort estimation

Florian Georg
florian.georg@ch.ibm.com

Jörg Möller
joerg.moeller@de.ibm.com

Stefan Hufnagl
hufnagl@de.ibm.com

IBM Software Group - Rational
AGENDA

> Agile Estimation... distributed?

> Solution Requirements

> Technologies
  – Rational Team Concert
  – Google Wave
  – Open Services for Lifecycle Collaboration

> Technical Solution Overview
  – Scrum Process Enactment
  – Wave Components
  – OSLC CM Protocol Usage

> Lessons learned, Q&A
Agile Estimation...

Planning Poker

- Common approach to estimate work items in agile projects
- Played with the team against backlog
- Relative (story points) or absolute (ideal days)
- 1+ rounds until consensus reached
- Improves common understanding of tasks and thus reduces risks
- Improves predictability and estimation quality
- Improves comparability over time

http://www.flickr.com/photos/11085191@N03/1073094268/
... distributed?

Collaborative game for co-located agile teams....

We have Offshoring with different Timezones

Small Teams skilled in agile practices

In IBM we have 40,000 developers

Many “traditional” teams, wanting to become more agile

How to scale Planning Poker and guide developers through the process?

Example: IBM Rational teams involved in RTC development
AGENDA

> Agile Estimation... distributed ?

> Solution Requirements

> Technologies
  – Rational Team Concert
  – Google Wave
  – Open Services for Lifecycle Collaboration

> Technical Solution Overview
  – Scrum Process Enactment
  – Wave Components
  – OSLC CM Protocol Usage

> Lessons learned, Q&A
What do we have?

Rational Team Concert (RTC)

Team development environment

Currently used by approx. 13,000 IBM developers

Used in distributed teams inside and outside IBM

Based on Eclipse and Jazz technology

Supports process enactment

Supports Agile iteration planning using work items (Scrum Process Template)

Story Work Items…

“Estimate” as an attribute…

…but no support to actually do estimating sessions!
What do we need?

Integration with collaboration tool for performing distributed Planning Poker

- Support both synchronous and asynchronous collaboration pattern
- Hidden estimates, until all participants have voted
- Team decides to commit.
- Team members may enter or leave the session at any time.
- Participants can vote/redraw votes at any time (until committed)
- Discussion should be auditable
AGENDA

> Agile Estimation... distributed?

> Solution Requirements

> Technologies
  - Rational Team Concert
  - Google Wave
  - Open Services for Lifecycle Collaboration

> Technical Solution Overview
  - Scrum Process Enactment
  - Wave Components
  - OSLC CM Protocol Usage

> Lessons learned, Q&A
Rational Team Concert

Integrated Team Collaboration Platform
- SCM
- Work Items
- Agile Planning
- Build Engine
- Process Enactment
- Dashboards, Reports

Why?
- Wide adoption inside IBM and customers
- Agile (Scrum) Process
- Integrated Planning
- Extensible (OSGi)

What?
- Work Items (Story)
- Agile Planning
- Process Advisor

Why?
- Agile (Scrum) Process
- Integrated Planning
- Extensible (OSGi)

What?
- Work Items (Story)
- Agile Planning
- Process Advisor

http://jazz.net
Plan structure in RTC

- Product Backlog
  - Release Backlog
  - Sprint 1 Plan
  - Sprint 2 Plans
- Team Backlog
  - Sprint 1 Plan
Google Wave

Online Collaboration and Communication
communication
Real-time
Rich media
discussions
Shared
document editing
Combines Chat, Mail, Forum metaphors

Why?
sync/async communication
(Robots, Gadgets)
Extensible
Open Protocol
Reviewable
(Playback)

What?
Gadget
Poker Robot
Google Wave: Robots and Gadgets (I)

Wave Robots

Wave Activity

document changes

modify/annotate Wave

„Active“ operations

oAuth

Ability

React

Profile Info

Event handler

Realized as GAE web application

wire protocol

JSON

Servlet

Servlet Mapping

Appengine-web.xml

AbstractRobot

„Servlet“

appengine-web.xml

Wave API libs

Servlet Mapping

(/_wave/)
Google Wave: Robots and Gadgets (II)

Wave Gadgets

JavaScript / DHTML / JSON

Private / shared state

objects

JSON data formats

(OpenSocial gadgets planned)

Embeddable HTML <div> + Javascript

Insert into Wave by URL

XML Document

Load special Google JS libraries

```xml
<Module>
<ModulePrefs title="State Example" height="120">
  <Require feature="wave" />
</ModulePrefs>
</Module>

<script type="text/javascript">
  var div = document.getElementById('content_div');

  function buttonClicked() {
    var value = parseInt(wave.getState().get('count', '0'));
    wave.setState().submitDelta({'count': value + 1});
  }

  function stateUpdated() {
    if (wave.getState().get('count')) {
      div.innerHTML = "The count is " + wave.getState().get('count');
    } else {
      div.innerHTML = "The count is " + wave.getState().get('count');
    }
  }

  function init() {
    if (wave.isInWaveContainer()) {
      wave.setStateCallback(stateUpdated);
    }
    gadgets.util.registerOnLoadHandler(init);
    // Reset value of "count" to 0
    function resetCounter() {
      wave.setState().submitDelta({'count': '0'});
    }
  }

  </script>

<input type="button" value="Click Me!" id="buttonCount" onClick="buttonClicked()">
<input type="button" value="Reset" id="buttonReset" onClick="resetCounter()">
</div>
</Content>
</Module>
```
Open Services for Lifecycle Collaboration

Common Development Lifecycle Services
Define common formats, services and protocols
Tool interoperability
RESTful inter
Industry coop
(IBM, Tasktop ...)

Why?
by RTC 2.0
for Wave

What?
OSLC CM 1.0 S
RTC specific for

http://open-services.net
What you can do with OSLC...

Discover and use RESTful CM services, e.g.
Creation Factory Service
Query Service
Embeddable UI Providers („Link Picker“)
...

OSLC specifies Formats, Services and Protocols
Common resource representations
(XML, JSON)
Services (Can/Must)
Service Discovery mechanism

Different OSLC specifications are currently developed
Change Management
Requirements Management
Architecture Management
...
AGENDA

> Agile Estimation... distributed?

> Solution Requirements

> Technologies
  – Rational Team Concert
  – Google Wave
  – Open Services for Lifecycle Collaboration

> Technical Solution Overview
  – Scrum Process Enactment
  – Wave Components
  – OSLC CM Protocol Usage

> Lessons learned, Q&A
- RTC Client/Server (with custom Plug-ins)
- Google Wave, using Robot and Gadget
- Robot and gadget Code hosted on AppEngine
- Robot talking „OSLC CM“ to RTC Server
Supported Process

1. Select Story Work Item
   - [change “Planned for” attribute]
   - [set state to “In Progress”]
   - e.g. drag story to Sprint Backlog

2. Story Points estimate set?
   - [No]
     - Reject change(s) to WI
   - [Yes]
     - Offer Quick-Fix Solution
     - Save Work Item
       - [No]
         - QuickFix selected?

3. Create Planning Poker Wave
   - [structured]
   - Create Wave
   - Add CardPicker Gadget
   - Invite Participants
   - Setup Poker Bot
   - Add Link to Work Item
   - Poker Wave created

4. Poker Wave created
   - Collaborate & Discuss
     - Participate in Wave
     - Submit a vote
       - [all participants have voted]
         - Reveal result
           - [play again]
     - [commit estimate]
       - Update Story WorkItem “Estimate” attribute
Scrum Process Enactment

RTC Plug-Ins

Precondition (server)

Fix (client)

Operations can be extended by custom pre-conditions

By project
  By team
  By role
  By Iteration

---

IBM Rational software
Process Enactment Plug-In (Server-side)

Server-side Plug-In (OSGI)

Provided by RTC SDK

Process specification (source)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.4">?
<plugin>
  <extension
    point="com.ibm.team.process.service.operationAdvisors">
    <operationAdvisor
      class="com.ibm.rational.rtwave.server.process.NeedEstimateOperationAdvisor"
      id="com.ibm.rational.rtwave.server.process.NeedEstimateOperationAdvisor"
      name="[RTC-Wave] Require team-estimate before assigning to Sprint"
      operationId="com.ibm.team.workitem.operation.workItemSave">
      <description>
        The team needs to agree on an estimate for the implementation effort,
        before you can assign this story to a Sprint backlog.
      </description>
    </operationAdvisor>
  </extension>
</plugin>

<behavior>
  <role id="default">
    <operation
      id="com.ibm.rational.rtwave.server.process.NeedEstimateOperationAdvisor"
      name="[RTC-Wave] Require team-estimate before assigning to Sprint"/>

    <preconditions>
      <precondition
description="The team needs to agree on an estimate for the implementation effort,
before you can assign this story to a Sprint backlog."/>
    </preconditions>
  </role>
</behavior>
```
Process Enactment Plug-In (Client-Side)

Client-Side Plug-In (Eclipse)

```
<extension
  point="com.ibm.team.process.client.advisorProblemResolutions">
  <advisorProblemResolution
    class="com.ibm.rational.rtcwave.client.process.StartWaveProblemResolution"
    description="Start a new Wave to collaborate and get a team-estimated effort"
    label="Start estimate session in Google Wave"
    problemType="com.ibm.rational.rtcwave.process.noEstimateProblem">
  </advisorProblemResolution>
</extension>
```

Generated by server-side Advisor report

Provided by RTC SDK
DEMO: Process Enactment
Wave Components: CardPicker Gadget

Track votes from participants
Calculate estimation results
Provide flippable „Card UI“
Notify Robot about commit state change

```xml
<Module>
  <ModulePrefs title="Jazz Planning Poker Card Picker"
               height="170" width="350">
    <Require Feature="wave" />
    <Require Feature="opensocial-0.8" />
    <Require Feature="minimessage" />
  </ModulePrefs>

  <UserPref name="debugmode" display_name="Debug Mode" datatype="bool"
            default_value="true"/>
</UserPref>

  <Content type="html" view="canvas">
    <![CDATA[
      <table>
        <tr><td id="content_div"></td></tr>
        <tr><td id="voters_list" style="font-size:9px"></td></tr>
        <tr>
          <td colspan="2"><button id="btnCommit" onClick="commit()"><img src="http://jazz-poker-bot.appspot.com/gadgets/card/CardPicker.js"></button>
          <button style="visibility: hidden" id="btnCommitSession" onClick="commitSession()"><img src="http://jazz-poker-bot.appspot.com/gadgets/card/CardPicker.js"></button>
          <button style="visibility: hidden" id="btnNewRound" onClick="newRound()"><img src="http://jazz-poker-bot.appspot.com/gadgets/card/CardPicker.js"></button>
        </td>
      </tr>
    ]]>>
  </Content>
</Module>
```
CardPicker Gadget Code

```javascript
function init() {
    wave.log("init");
    if (wave && wave.isInWaveContainer()) {
        preload();
        wave.setStateCallback(stateUpdated);
        wave.setParticipantCallback(stateUpdated);
    }
}

debug("Registering onLoadHandler...");
gadgets.util.registerOnLoadHandler(init);

function stateUpdated() {
    debug("stateUpdated");
    if (!wave.getState()) {
        return;
    }

    // init to 'unsure'
    if (!_initialized) {
        _initialized = true;
        showCard(-1);
    }

    // compute results
    computeResults();
}

// show list of voted users
var userList = "";
var allParticipants = wave.getParticipants();
for (var i = 0; i < allParticipants.length; i++) {
    var participant = allParticipants[i];
    var id = participant.getId();
    var t = "";
    //info(votedUsers.join("",)): 
    if (votedUsers.join("",).indexOf(id) < 0) {
        t = "not";
    }
    userList += "<img src="/image_base"/"+t+"voted.png'/>";
    user_id = "" + participant.getDisplayName() + "<br/>";
    //userlist += "<img src="/"+ participant.getThumbnailUrl() + "'/">";
    // + participant.getId() + "<br/>
    document.getElementById(VOTERS_DIV).innerHTML = userList;
    adjustHeight();
}
```

shared gadget state (key/value pairs)
calculate private and "shared" data
update UI
DEMO: CardPicker Gadget
Wave Component: PokerBot

Implemented in Java
Hosted on GAE
Python also possible

Active Robot
Allowed to create new Waves
OAuth Security

“Custom” HTTP POST interface
RESTful Wave Factory
Called from RTC client
OSLC Change Management - Usage

Used for updating Story Work Item from Robot

Robot reacts on Gadget „commit“ event

RTC Simple Query Service

GET XML representation

Update „estimate“ attribute

PUT XML representation

Need to deal with RTC auth, HTTP specifics, OSLC protocol & formats
Update Story Work Item

Send back (partial) representation of Work Item

PUT HTTP Request

ETag to detect concurrent modifications

```
/0J39QJu-Ed6cerS91b5AWw?oslc_cm.properties=rtc_cm:estimate
Accept: application/x-oslc-cm-change-request+xml
If-Match: _1am9cFm0Ed6ELJg2MQ68Kg
Content-Type: application/x-oslc-cm-change-request+xml

<oslc_cm:ChangeRequest
xmlns:oslc_cm="http://open-services.net/xmlns/cm/1.0/"

<rtc_cm:estimate>13</rtc_cm:estimate>

</oslc_cm:ChangeRequest>
```
Demo: OSLC Update
AGENDA

> Agile Estimation... distributed?

> Solution Requirements

> Technologies
  – Rational Team Concert
  – Google Wave
  – Open Services for Lifecycle Collaboration

> Technical Solution Overview
  – Scrum Process Enactment
  – Wave Components
  – OSLC CM Protocol Usage

> Lessons learned, Q&A
Lessons Learned

- OSGi, REST, and OSLC offer a great approach for heterogenous tool integration and extensibility
- OSLC spec is very generic and basic (e.g. „Estimate“ attribute not standard)
- Google changes beta API and running systems quite often (breaks working code)
- No official client lib to simplify OSLC usage (we're working on it)
- Google Wave still in early stages, hard to debug robots and gadgets (no local install)
- RTC mature, but SDK takes some time to setup
- Google documentation is much better, RTC SDK needs more „learning from code“
- OSLC XML formats are quite verbose; take a closer look at RDF
- Be aware of AppEngine restrictions (e.g. self-signed HTTPS/SSL certs)
Links

- [http://jazz.net/wiki/bin/view/Main/RtcSdk20](http://jazz.net/wiki/bin/view/Main/RtcSdk20) – RTC 2.0 SDK (Wiki)

- [http://wave.google.com](http://wave.google.com) – Everything about Google Wave

- [http://open-services.net](http://open-services.net) – Collaborative Wiki and Whitepapers for OSLC
- [http://open-services.net/bin/view/Main/CmSpecificationV1](http://open-services.net/bin/view/Main/CmSpecificationV1) – OSLC Specification for Change Management, Version 1
- [http://jazz.net/library/article/352](http://jazz.net/library/article/352) – How to consume the Rational Team Concert change management services
How do we want to play?

The “Playing Poker” Process

- Started manually or triggered by Process
- IDE integration offers to create & link to a new Poker Session
- Session is about informal communication and voting from all team members
- Team commits results back into RTC, e.g. update the story estimation attribute.