DSDP Device Debugging Project Update

by Pawel Piech
Wind River Systems, Inc.
What is it?

- **Goal**
  - To improve the overall experience of device software debugging in Eclipse by influencing other projects and filling in missing functionality.

- **DD Project as a forum**
  - Diverse participation: (Wind River (lead), IBM, Mentor Graphics, Nokia, PalmSource, TI, QNX, Freescale, Ericsson, ARM)
  - Close collaboration with other projects (Platform, CDT, etc).
Debugger Views

To allow complete customization of contents of Eclipse's standard debugger data views (Debug, Variables, Registers) and to create an API for populating these views that makes minimal assumptions about the structure and format for this debug data.
Debugger Views - Flexible Hierarchy
Debugger Views - Flexible Hierarchy

- Adapter Types – Each adapter provides a property for each element:
  - IElementContentProvider – children
  - IElementLabelProvider – text, icon, font, color for each column for an element
  - IModelProxy – model event handler, translates events into view update requests
  - IColumnPresentation – list of columns
  - IElementEditor – a modifier and cell editors for each column
  - IElementMementoProvider – serializable data
  - IViewerInputProvider – proxy input into a viewer
Debugger Views - Flexible Hierarchy

1 (create)
2 update(...)
3 Model specific API
4 setValues(..)
5 done()
6 setData()
Memory View

Provider memory view support suitable for Embedded development

Support for custom renderings (Eclipse 3.1)

Bugfixes to support Traditional Rendering Callisto (Eclipse 3.2)

Concept presented

Prototype checked into HEAD branch and presented

Bug fixes

Custom Renderings Support (Eclipse/Platform/Debug)

Traditional Rendering (Project/Sub-Project/Component)

2005 2006 2007 2008

Europa (DD 0.9)

Ganymede (DD 1.0)
Memory View - Traditional Memory Rendering
SPIRIT Debug Working Group

To achieve some level of standardization of "target descriptions" used by device debuggers in Eclipse

- IP-XACT Editor (DSDP/D D/IP-XACT)
  - SPIRIT Debug working group starts to participate in DD
  - Lengthy legal review of SPIRIT and Eclipse licenses
  - IP-XACT editor contributed to DD
  - Europa (DD 0.9)
  - Support for IP-XACT 1.4 (Planned)

- Debugger D at a Schema
  - Community voice of support for standardization effort

- Views based on IP-XACT
  - IP-XACT driven register view plugin submission
  - SPIRIT Debug WG gathering requirements for schema

Timeline:
- 2005
- 2006
- 2007
- 2008
- Now
DSF (Debugger Services Framework)

Create an alternative debug model API that will accommodate diverse needs of embedded debuggers: performance, modularity, extensibility.

DSF
(DSD P/D D/D SF)

- Design proposal presented
- Prototype checked into CVS
- Europa (DD 0.9)
- Continued development to support GDB/MI
- DSF included in Wind River's commercial product
- Ganymede (DD 1.0)

2005 2006 2007 2008

Now
DSF – Concurrency Model

- All public service APIs should be accessed on a single “session” thread
- The session thread is managed by an executor object associated with the session
- The session thread acts as a global lock for state data accessible through public APIs of all the services at the session
- Services are still free to create separate worker threads to execute long-running operations
- This is the same model as one used by SWT and most other window toolkits
DSF – Asynchronous Interfaces

Client

1 (create)

RequestMonitor

run() {
    Client's code
}

Service

request(RequestMonitor)

done() (create)

RequestMonitor

run() {
    Service's code
}

Another Service

anotherRequest(RequestMonitor)

done() (create)

RequestMonitor

run() {
    Another Service's code
}

Yet Another Service

yetAnotherRequest(RequestMonitor)

done() (create)

...
DSF – Data Model

- Services' data handles implement **IDMContext** interface
- Contexts are immutable, light-weight, and must properly implement `equals()` and `hashCode()`.
- A service can build on another service's context object to provide additional data
- Contexts are equal if all the contexts that they build on are equal
- Services accept generic contexts as arguments and search the context hierarchy for the relevant handle to act upon
GDB/MI Reference Implementation

Create a GDB-based debugger which implements DSF model APIs. This debugger should be functionally equivalent to the GDB debugger using CDI and standard debug model.

GDB/MI Reference Implementation (DSDP/DD/GDB)

- Limited prototype checked into CVS along with DSF
- Europa 0.9
- CDT to evaluate using DSF-GDB in Ganymede
- Ericsson contributes resources to project
- Ganymede (DD 1.0)

2005 2006 2007 2008

Now
Disassembly

To provide a disassembly editor and replace existing CDT disassembly view.

Disassembly Editor
(Tools/CDT/Debug)

Disassembly requirements gathered

Ganymede
(CDT 5.0)

Refactoring and adopting to an open API

2005
2006
2007
2008
Now
Multi-Context

To improve workflows and context switching when debugging multiple threads, processes, targets, etc.

Pin & Clone
(Eclipse/Platform/Debug)

“Pin and Clone” for debug views workflow proposed

Patch contributed to Platform, but not used in 3.3

Refined proposal for managing multiple view instances

Ganymede (Eclipse 3.4)

Contribute patches to Platform project.

Multi-Context
(Eclipse/Platform/Debug)