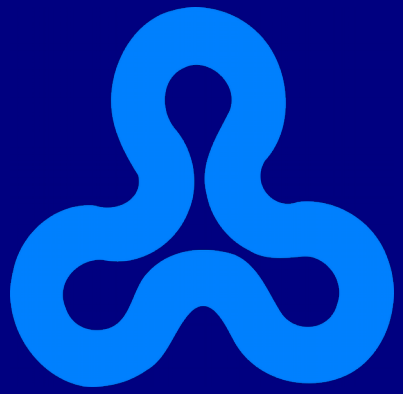
Language Oriented Modularity: From Theory to Practice

Arik Hadas
Dept. of Mathematics and Computer Science
The Open University of Israel



Joint Work With:
David H. Lorenz

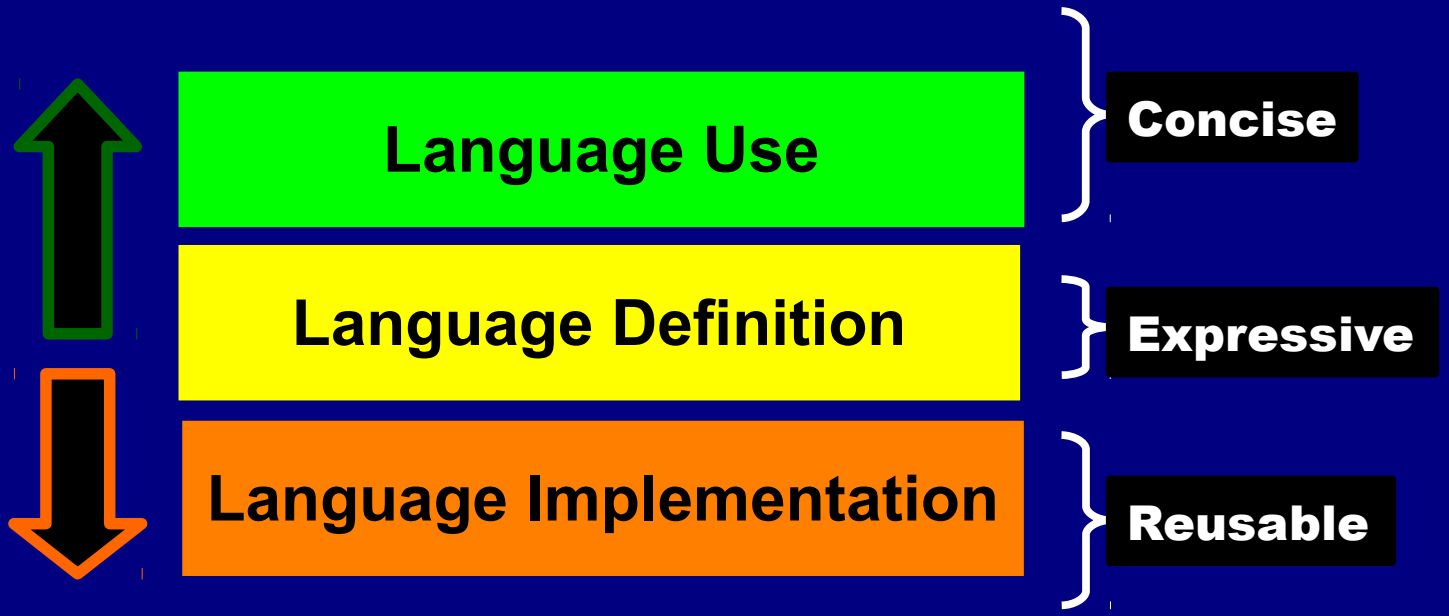
Language Oriented Modularity (LOM)

LOM is a programming methodology that puts DSALs at the center of the software modularization process

- **Traditional Process**
  - **Modularize your *concerns* with an (aspect) *programming language***
- **The LOM Process**
  - **Design (domain specific aspect) *programming languages* for your *concerns***

The Language Oriented Process

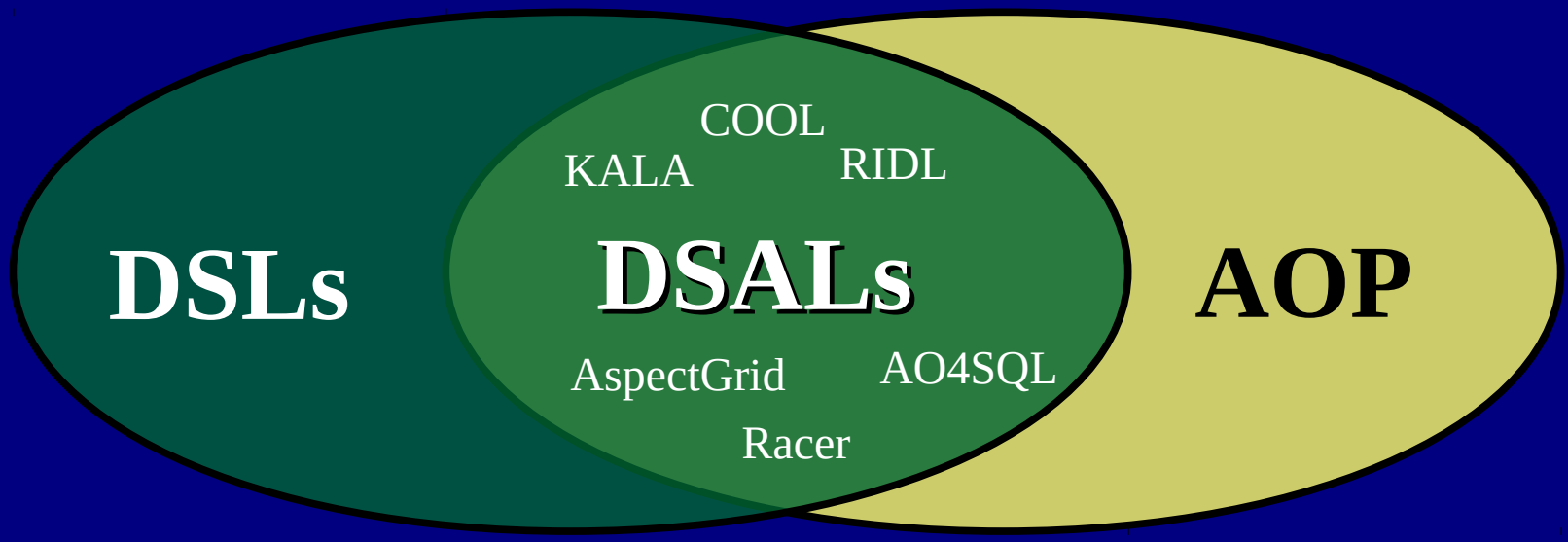
- **Works middle-out**
  - Languages are tailored to the problem at hand



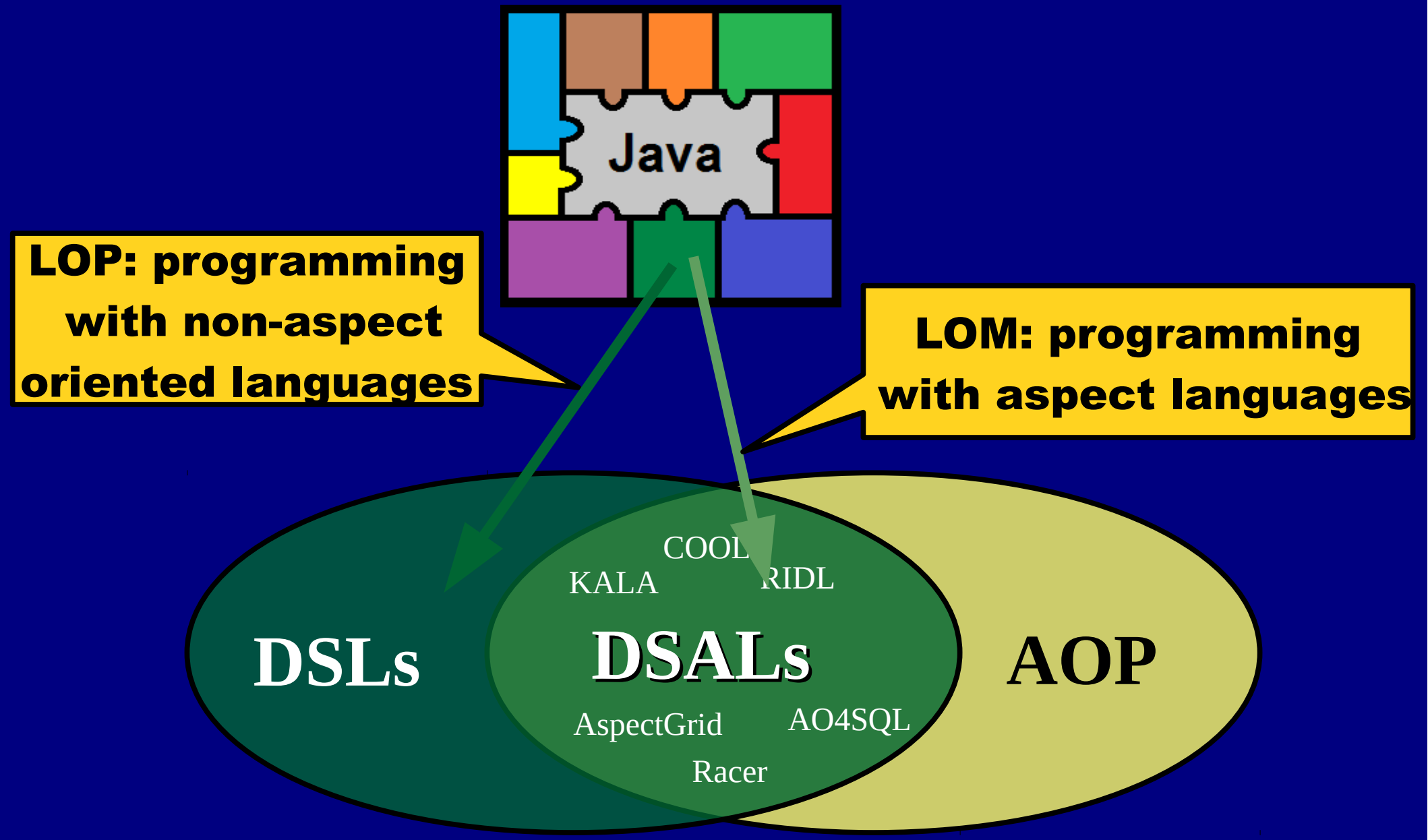
- **Similar to LOP**
  - But with DSALs

DSALs

- **Domain Specific**
  - Domain-specific abstractions
- **Aspect Oriented**
  - Modularization (weaving) mechanism



In Principle: LOM is similar to LOP



In Practice: LOM is not Cost-Effective

- **LOM is more costly than LOP**
  - DSALs are more complex to implement
- **LOM is less effective than LOP**
  - The ease of using DSALs (w.r.t. GPALs) is lower than the ease using DSLs (w.r.t. GPLs)

	DSL	DSAL
Implementation cost		
Effective programming		

In Practice: LOM is ~~not~~ Cost-Effective

- **LOM is more costly tha**
  - DSALs are more complex
- **LOM is less effective t**
  - The ease of using DSALs (w.r.t. GPALs) is lower than the ease using DSLs (w.r.t. GPLs)

This Work:
Making LOM Practical

	DSL	DSAL
Implementation cost		
Effective programming		

Outline

- **Motivation**
- Problem
- Approach
- Validation
- Evaluation

The Need for DSALs

- **Separation of crosscutting concerns**
  - Often the right tool for crosscutting concerns found in modern software projects
- **Domain-specific abstractions and notations**
  - Simpler than general-purpose aspect languages

	DSL	GPAL	DSAL
Improved Modularity			
Declarative & Simple Syntax			

Why *On-Demand* DSALs?

- DSALs tend to be tightly coupled with the app
  - Depend on the code structure
  - Depend on the representation of data
- Less reusable across apps
  - DSALs are typically application-specific

	DSL	DSAL
Reusable Across Applications		

DSL Reuse

Application 1

```
public void logName(Guid id) {
    String name = dao.getName(id);
    logger.log("user " + id + ": " + name);
}
```

Application 2

```
public void foo(..) {
    // … skipped ...
    /* … */ dao.getName(id);
    // … skipped …
}
```

DSL code

```
SELECT name FROM users WHERE user_id=<id>;
```

DSL can be reused across applications

DSL Reuse

Application 1

```
public void logName(Guid id) {
    String name = dao.getName(id);
    logger.log("user " + id + ": " + name);
}
```

Application 2

```
public void foo(..) {
    // … skipped ...
    /* … */ dao.getName(id);
    // … skipped …
}
```

DSL code

```
SELECT name FROM users WHERE user_id=<id>;
```

DSL can be reused across applications

Aspect in a DSAL called muAudit

jobs.audit

```
1  logs for com.mucommander.job.impl.CopyJob:
2   case start log COPY_STARTED with nbFiles baseSourceFolder baseDest
3   case finish log COPY_FINISHED with nbFiles baseSourceFolder baseDe
4   case interrupt log COPY_INTERRUPTED with baseSourceFolder baseDest
5   case pause log COPY_PAUSED with baseSourceFolder baseDestFolder nb
6   case resume log COPY_RESUMED with baseSourceFolder baseDestFolder
7  ;
8
9  logs for com.mucommander.job.impl.MkdirJob:
10  case start & mkfileMode log MKFile_STARTED with files
11  case start log MKDIR_STARTED with files
12  case finish & mkfileMode log MKFile_FINISHED with files
13  case finish log MKDIR_FINISHED with files
14  case interrupt & mkfileMode log MKFile_INTERRUPTED with files
15  case interrupt log MKDIR_INTERRUPTED with files
16  case pause & mkfileMode log MKFile_PAUSED with files
17  case pause log MKDIR_PAUSED with files
18  case resume & mkfileMode log MKFile_RESUMED with files
19  case resume log MKDIR_RESUMED with files
20 ;
```

Aspect in a DSAL called muAudit

jobs.audit

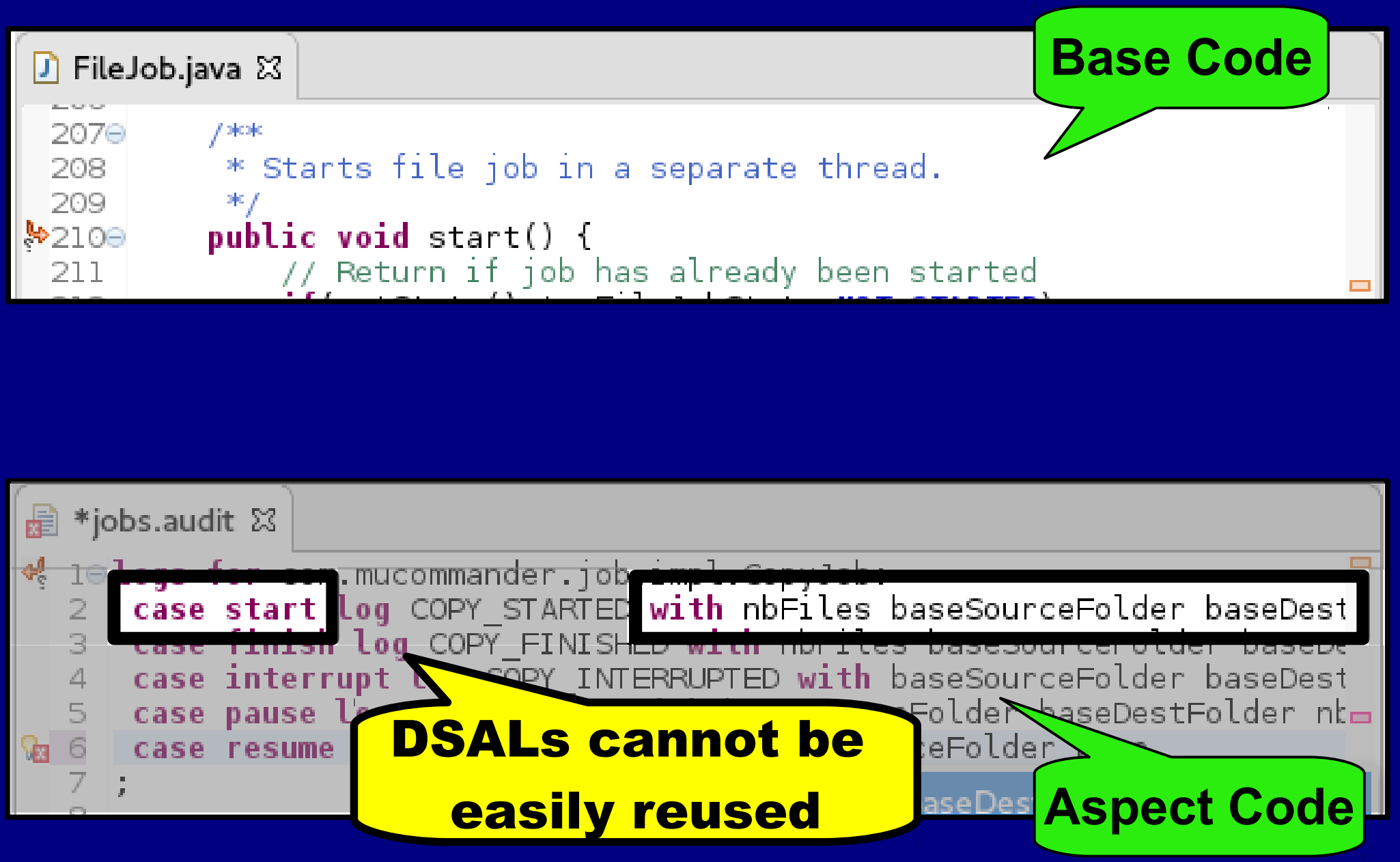
```
logs for com.mucommander.job.impl.CopyJob:
 case start log COPY_STARTED with nbFiles baseSourceFolder baseDest
 case finish log COPY_FINISHED with nbFiles baseSourceFolder baseDe
 case interrupt log COPY_INTERRUPTED with baseSourceFolder baseDest
 case pause log COPY_PAUSED with baseSourceFolder baseDestFolder nb
 case resume log COPY_RESUMED with baseSourceFolder baseDestFolder
;

logs for com.mucommander.job.impl.MkdirJob:
 case start & mkfileMode log MKFile_STARTED with files
 case start log MKDIR_STARTED with files
 case finish & mkfileMode log MKFile_FINISHED with files
 case finish log MKDIR_FINISHED with files
 case interrupt & mkfileMode log MKFile_INTERRUPTED with files
 case interrupt log MKDIR_INTERRUPTED with files
 case pause & mkfileMode log MKFile_PAUSED with files
 case pause log MKDIR_PAUSED with files
 case resume & mkfileMode log MKFile_RESUMED with files
 case resume log MKDIR_RESUMED with files
;
```

The Method that Starts File Jobs

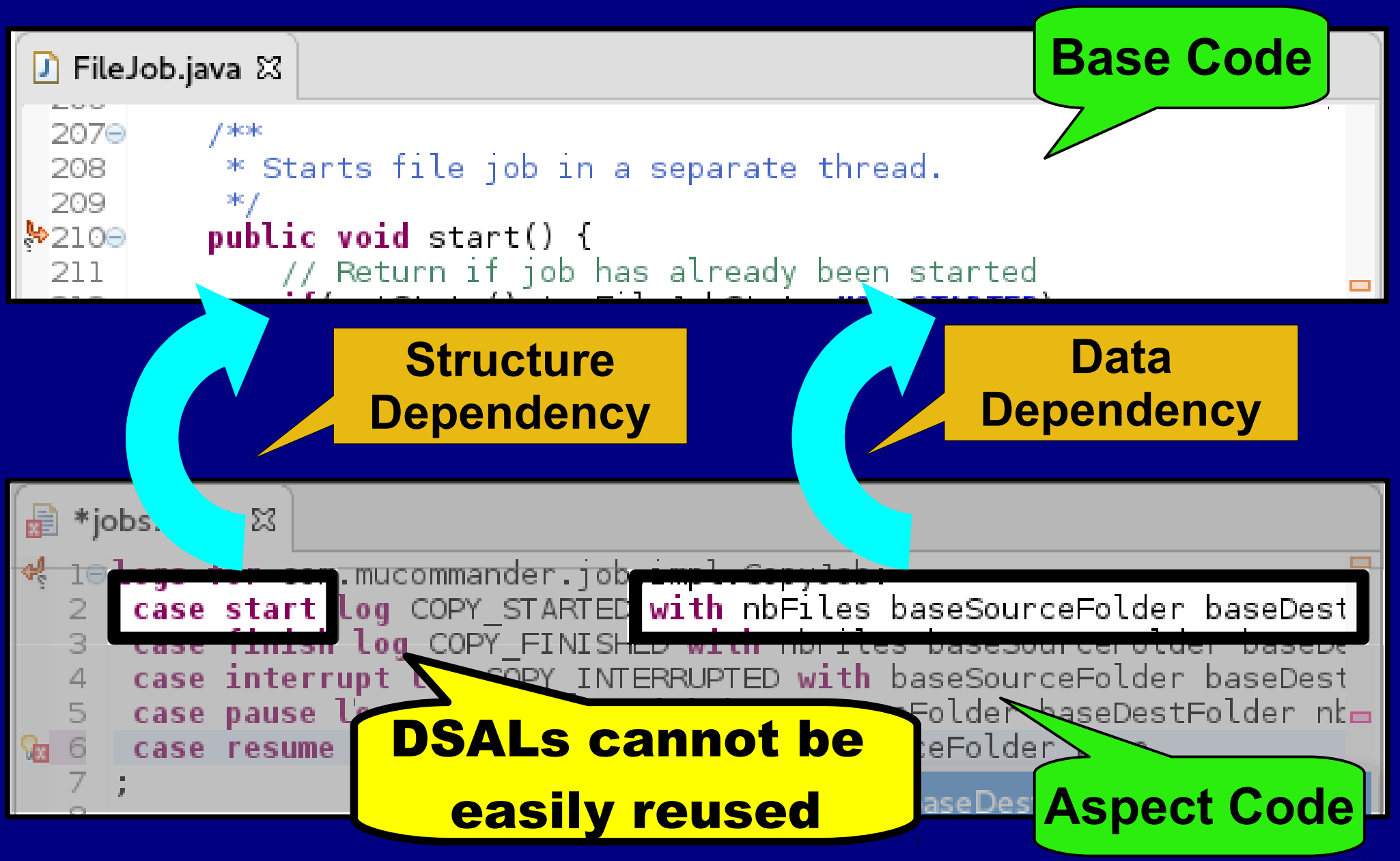
FileJob.java

```
207     /**
208      * Starts file job in a separate thread.
209      */
210     public void start() {
211         // Return if job has already been started
212         if(getState() != FileJobState.NOT_STARTED)
213             return;
214
215         // Pause auto-refresh during file job as it potentially mo
216         // and would potentially cause folder panel to auto-refres
217         getMainFrame().getLeftPanel().getFolderChangeMonitor().set
218         getMainFrame().getRightPanel().getFolderChangeMonitor().se
219
220         setState(FileJobState.RUNNING);
221         startDate = System.currentTimeMillis();
222
223         jobThread = new Thread(this, getClass().getName());
224         jobThread.start();
225     }
226
```

Coupling of DSALs with the Base Code



Coupling of DSALs with the Base Code



The Need for LOM

- The need for application specific DSALs calls for LOM
  - Having LOM for DSALs is even more crucial than having LOP for DSLs
- The main obstacle
  - Cost-effectiveness

Outline

- Motivation
- **Problem**
- Approach
- Validation
- Evaluation

DSALs are Second-Class

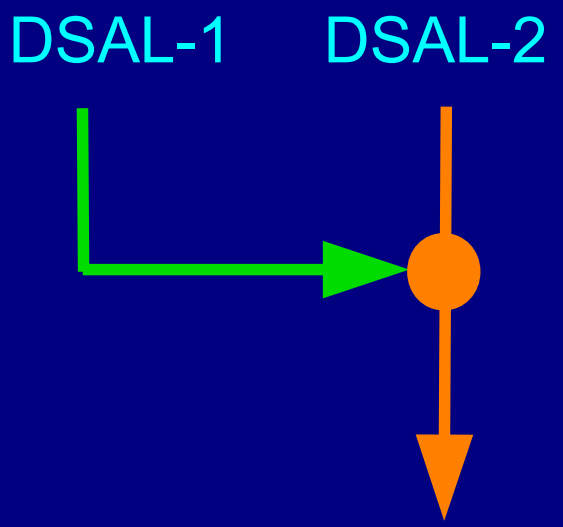
- **Second-class DSLs**
  - More costly than ordinary DSLs
- **Second-class aspect languages**
  - Less effective than ordinary DSLs
- **Results in poor cost-effectiveness of LOM**

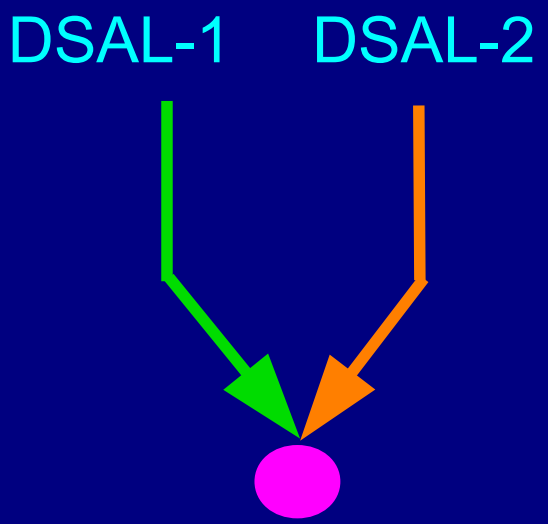
Why DSALs are Second-Class DSLs

- Language workbenches ease DSLs creation
  - Produce a parser for the custom syntax
  - Produce a transformation to some GPL
- But LWs do not help with creation of DSALs
  - Transformations of DSALs typically do not preserve the join-point “finger print” (not semantic-preserving)
  - Without a transformation, the weaving semantics are hard to implement

Multi-DSAL Conflicts

- **Foreign Advising**
  - Advise written in one DSAL advises a join point located within an aspect written in a different DSAL
- **Co-Advising**
  - Multiple pieces of advice written in different DSALs advise the same join point within the base code





Why DSALs are 2nd-Class Aspect Langs

- DSALs lack development tools
  - Thus complex to edit, browse and compile
- Hinders effective use of DSALs
  - Despite being simple and declarative

	GPALs	DSALs
Common Editing Tools		
Aspect Development Tools		
Build Tools		

Tool Support for DSALs

This Work

Language	Support	CF	LW+ GPAL	LW+ CF	Practical LOM
Domain-specific	*DSAL interoperability*	✓		✓	✓
	Development process		✓		✓
Aspect-oriented	*Editing tools*		✓	✓	✓
	Aspect development tools				✓
	Compilation				✓

CF = aspect Composition Framework

LW = Language Workbench

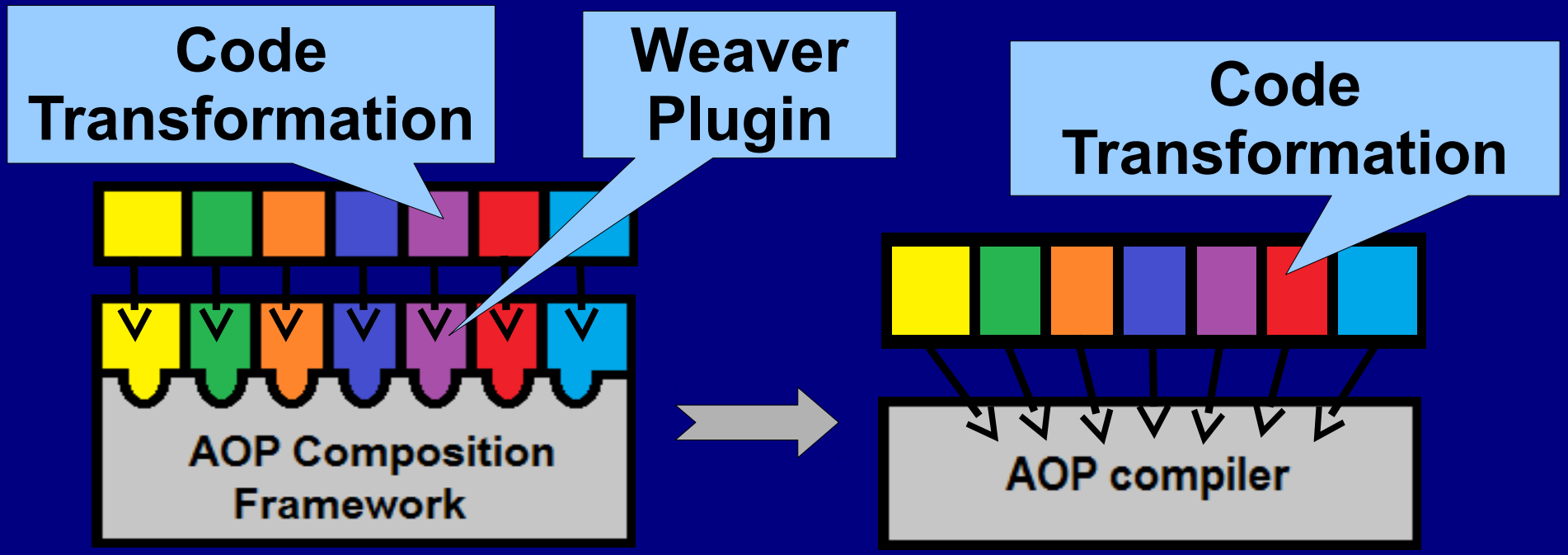
GPAL = General Purpose Aspect Language

Outline

- Motivation
- Problem
- **Approach**
- Validation
- Evaluation

Key Idea

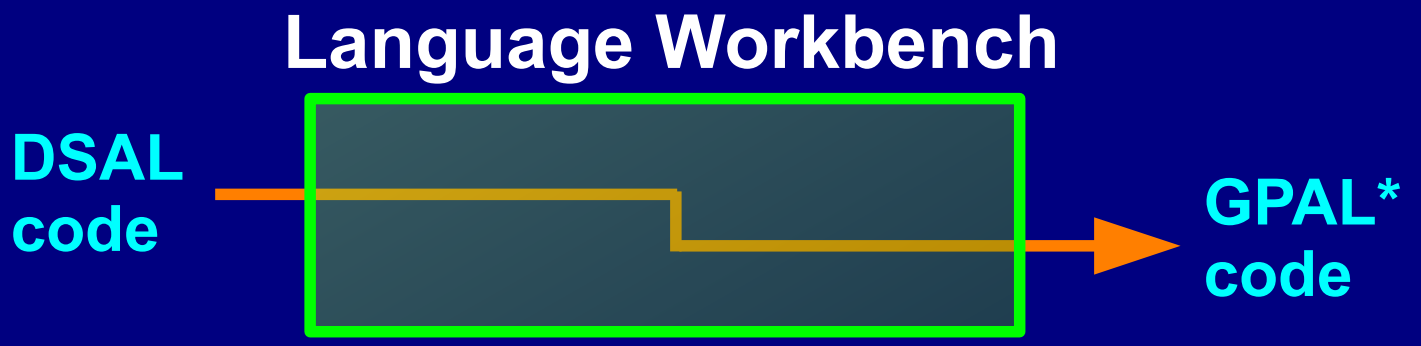
- **Transformation of DSALs into a GPAL annotated with metadata**
  - DSL-like development process for DSALs
  - The use of tools available for the GPAL



(1) Metadata for Handling Multi-DSAL Conflicts

- **Hide join points from other DSALs**
  - Forgoing complete obliviousness to prevent foreign advising conflicts
- **Define advice-level ordering values**
  - Fine-grained advice ordering to prevent co-advising conflicts

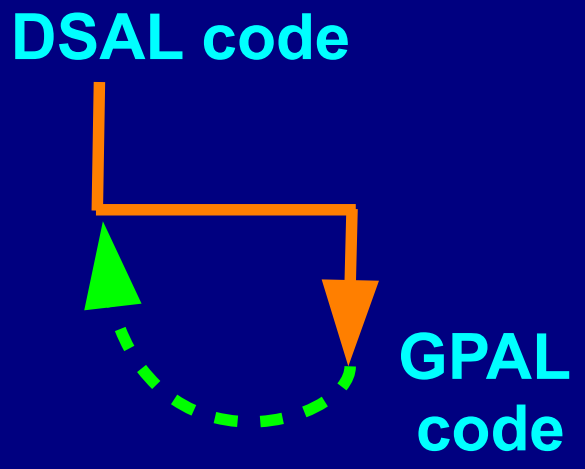
First-Class Equality with DSLs

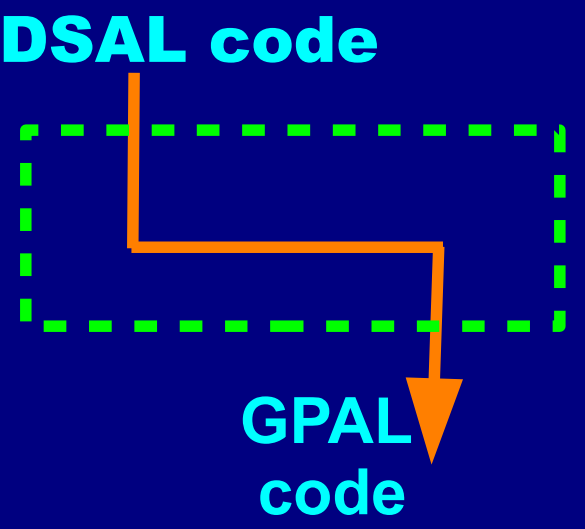


- DSL-like Implementation process for DSALs
  - Parsing the DSAL code
  - Transforming DSAL code to a GPAL* code
  - No compiler (weaver) modification
- DSL-like development tools for DSALs
  - Leverage supportive tools by an existing language workbench

(2) Metadata for Compatibility with GPAL Tools

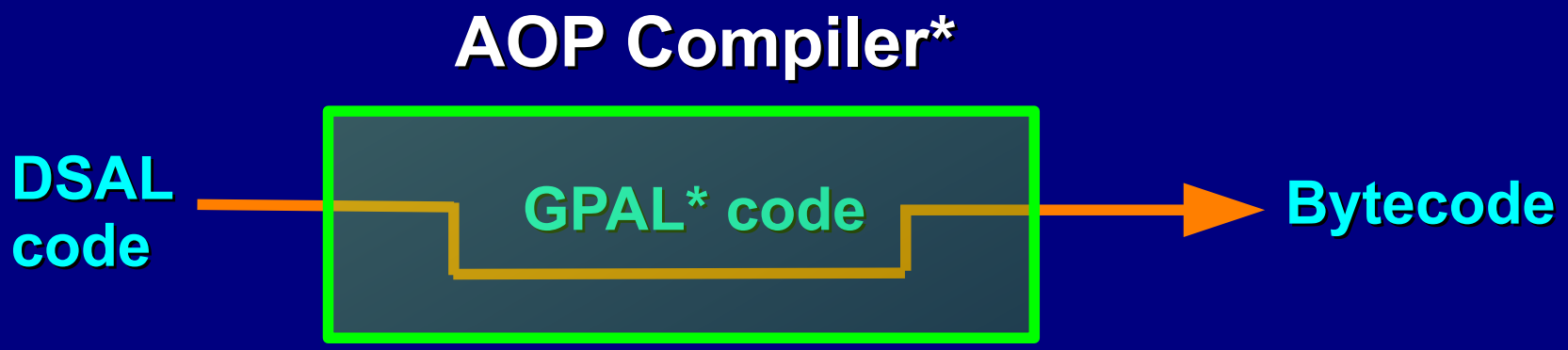
- Preserve source code location of advice during the transformation
  - To enable browsing and navigation using GPAL tools
- Produce transformations that can be invoked internally by the compiler
  - To enable compilation using GPAL tools





First-Class Equality with Aspect Langs

- Leverage aspect development tools for GPAL
  - Provide browsing and navigation for DSALs
- Leverage compilation tools for the GPAL
  - Enable to build the app the same way with DSALs
- The language workbench produces IDE plugin
  - Providing editing tools for DSALs



Outline

- Motivation
- Problem
- Approach
- **Validation**
- Evaluation

Extending AspectJ with Metadata

- We extended AspectJ
  - Small set of annotations and interface
- We modified the ajc compiler
  - One-time effort
  - Minimal and optional modifications in case they need to be redone in future releases
- We modified AJDT
  - Minor adjustment

@Hide

Suppress join-points associated with a particular program element

```
@Target(ElementType.TYPE)
public @interface HideType {
    TypeJoinpoint[] joinpoints() default { TypeJoinpoint.PRE_INIT,
       TypeJoinpoint.INIT, TypeJoinpoint.STATIC_INIT,
       TypeJoinpoint.WITHIN_INIT, TypeJoinpoint.WITHIN_STATIC_INIT };
}
```

```
@Target(ElementType.METHOD)
public @interface HideMethod {
    MethodJoinpoint[] joinpoints() default { MethodJoinpoint.CALL,
                MethodJoinpoint.EXECUTION, MethodJoinpoint.WITHIN };
}
```

```
@Target(ElementType.FIELD)
public @interface HideField {
    FieldJoinpoint[] joinpoints() default { FieldJoinpoint.SET,
                                            FieldJoinpoint.GET };
}
```

@Order

```
public @interface Order {
    double value();
}
```

Order advice according to the value of the @Order annotation

@BridgedSourceLocation

Store the original location of advice in the generated code

```
public @interface BridgedSourceLocation {
    public String file();
    public int line();
    public String module();
}
```

AJDT uses the location pointed to by this annotation, if it exists, as the source location of advise

Transformation

```
public interface Transformation {
   String extension();
   File convert2java(File input)
                              throws Exception;
}
```

The compiler uses this interface to identify which files should be transformed (using the extension method) and to transform them internally (convert2java)

Outline

- Motivation
- Problem
- Approach
- Validation
- **Evaluation**

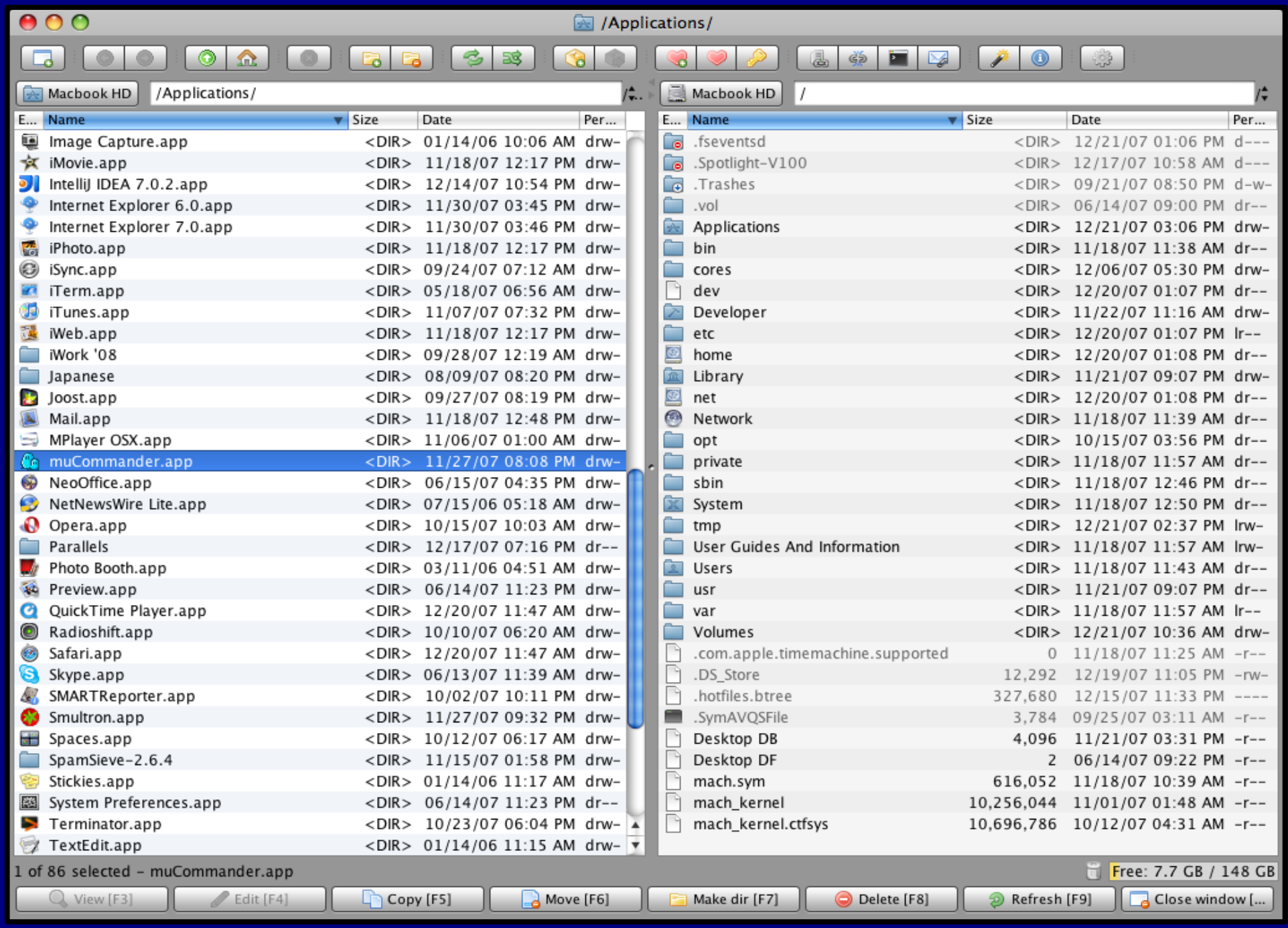
Evaluation

- Experimental evaluation
  - Implementing new crosscutting feature in the *muCommander* project
  - Separating existing crosscutting concerns in the *oVirt* project
- Comparative evaluation
  - Comparing the implementation of COOL with its implementation in AWESOME

Case Study I: LOM for muCommander

- We applied LOM to muCommander
  - Implemented a new DSAL for an auditing named *muAudit*
  - Implemented aspect solution for auditing of two file operations

About muCommander

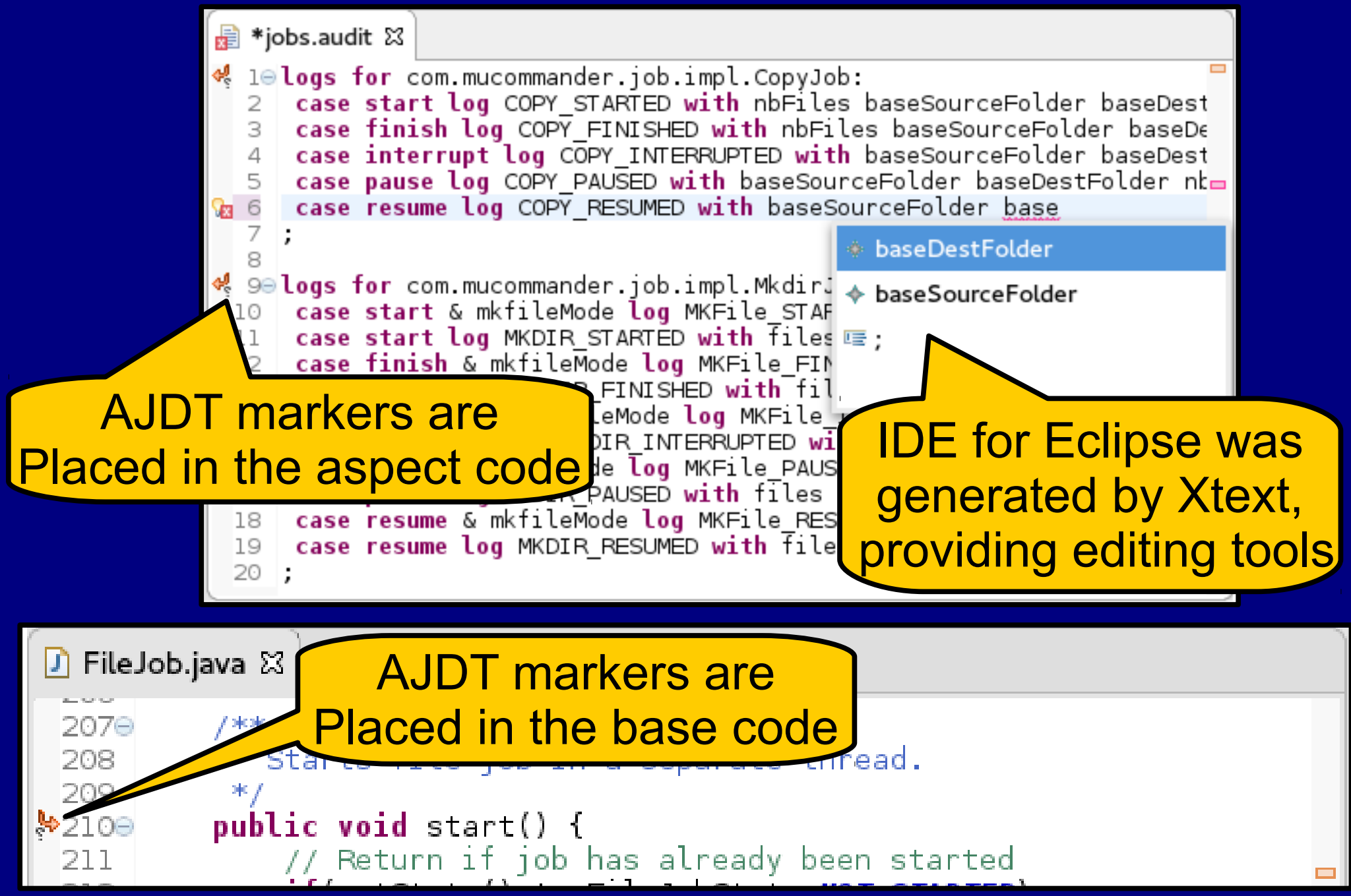


Language Definition

```
Model: (commands+=Command)*;

Command:
  'logs for' type=[types::JvmDeclaredType|QualifiedName] ':' (cases+=Case)* ';'
;

Case:
  'case' state=State ('&' (fields+=[types::JvmField]))*
  'log' msg=[types::JvmEnumerationLiteral] ('with' (vars+=[types::JvmField])+)?
;

enum State: start | finish | interrupt | pause | resume;

QualifiedName: ID ("." ID)*;
```

Defined in the language grammar definition of Xtext

Language Use



Language Implementation

```
logs for com.mucommander.job.impl.CopyJob:
    case start log COPY_STARTED with
        nbFiles baseSourceFolder baseDestFolder files
    case finish log COPY_FINISHED with
        nbFiles baseSourceFolder baseDestFolder
    case interrupt log COPY_INTERRUPTED with
        baseSourceFolder baseDestFolder
    case pause log COPY_PAUSED with
        baseSourceFolder baseDestFolder nbProcessedFiles
    case resume log COPY_RESUMED with
        baseSourceFolder baseDestFo
```

Transformation with Xtend

@Hide

@BridgedSourceLocation

```
@HideType
public privileged aspect Logs {
  @BridgedSourceLocation(line=1,
      file="/mucommander/src/main/java/ ... /jobs.audit",
      module="jobs.audit")
  after(CopyJob job): execution(void start()) && this(job) {
    if (true) {
      audit("start copying {0} files from {1} to {2} ({3})",
          job.nbFiles, job.baseSourceFolder,
          job.baseDestFolder, job.files);
      return;
    }
  }
  //... skipped ...
}
```

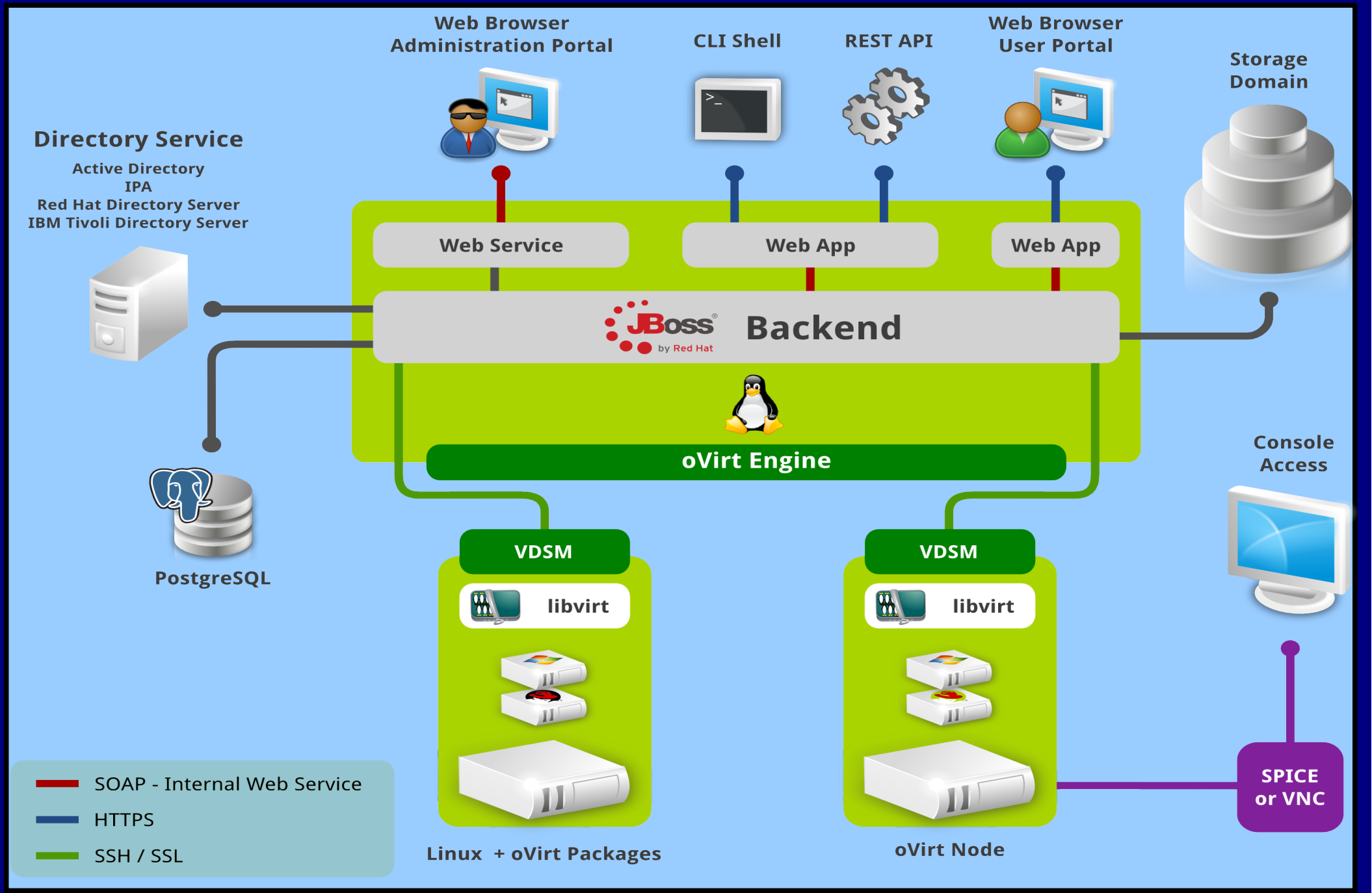
Lessons from Case Study I

- Implementation in Xtext
  - Complete implementation in an existing language workbench, like that of a DSL
- Development tools for programming with muAudit, like those available for a GPAL
  - Editing tools and aspect dev. tools
  - The project is compiled as if using AspectJ
- Cost-effective LOM process for a new crosscutting feature (compared to LOP)

Case Study II: LOM for oVirt

- We implemented DSALs for 3 crosscutting concerns found in the oVirt project
  - Synchronization
  - Permission checks
  - Auditing

About oVirt


Web Browser
Administration Portal
CLI Shell
REST API
Web Browser
User Portal
Storage
Domain
Directory Service
Active Directory
IPA
Red Hat Directory Server
IBM Tivoli Directory Server
Web Service
Web App
Web App
JBoss
by Red Hat
Backend
oVirt Engine
Console
Access
PostgreSQL
VDSM
libvirt
VDSM
libvirt
SPICE
or VNC
SOAP - Internal Web Service
HTTPS
SSH / SSL
Linux + oVirt Packages
oVirt Node

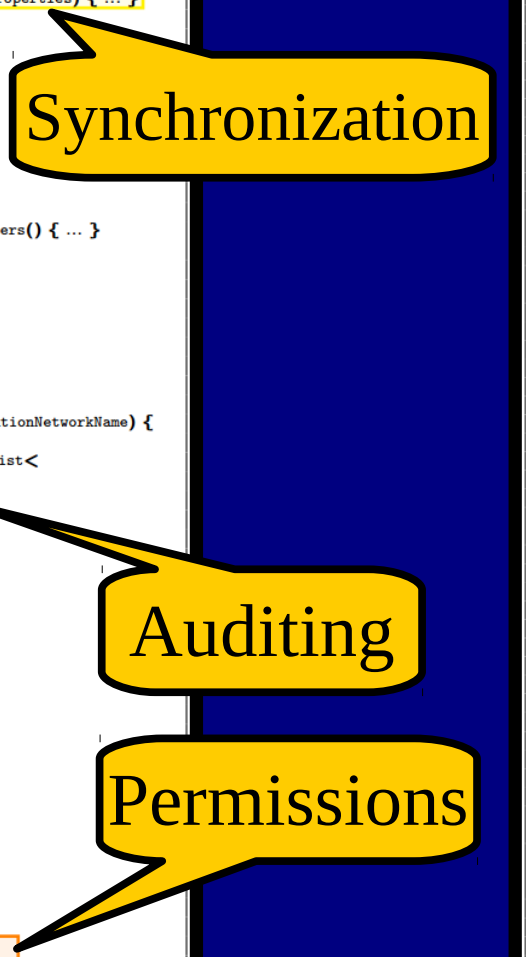
Scattered Code in oVirt

MigrateVmCommand

```
public class MigrateVmCommand<T extends MigrateVmParameters> ... {
    private VDS destinationVds;
    private EngineError migrationErrorCode;
    private Integer actualDowntime;
    public MigrateVmCommand(T parameters) { ... }
    public MigrateVmCommand(T migrateVmParameters, CommandContext cmdContext) { ... }
    @Override
    protected LockProperties applyLockProperties(LockProperties lockProperties) { ... }
    public final String getDestinationVdsName() { ... }
    public String getDueToMigrationError() { ... }
    protected VDS getDestinationVds() { ... }
    @Override
    protected void processVmOnDown() { ... }
    protected boolean initVdss() { ... }
    private List<Guid> getDestinationHostList() { ... }
    @Override
    protected void executeVmCommand() { ... }
    private boolean perform() { ... }
    private boolean migrateVm() { ... }
    private MigrateVDSCommandParameters createMigrateVDSCommandParameters() { ... }
    @Override
    public void runningSucceded() { ... }
    protected void getDowntime() { ... }
    private void updateVmAfterMigrationToDifferentCluster() { ... }
    private Boolean getAutoConverge() { ... }
    private Boolean getMigrateCompressed() { ... }
    private int getMaximumMigrationDowntime() { ... }
    private boolean isTunnelMigrationUsed() { ... }
    private String getMigrationNetworkIp() { ... }
    private String getMigrationNetworkAddress(Guid hostId, String migrationNetworkName) {
        ... }
    protected boolean migrationInterfaceUp(VdsNetworkInterface nic, List<
        VdsNetworkInterface> nics) { ... }
    @Override
    public AuditLogType getAuditLogTypeValue() { ... }
    private AuditLogType getAuditLogForMigrationStarted() { ... }
    protected AuditLogType getAuditLogForMigrationFailure() { ... }
    protected Guid getDestinationVdsId() { ... }
    protected void setDestinationVdsId(Guid vdsId) { ... }
    @Override
    protected boolean canDoAction() { ... }
    protected void setActionMessageParameters() { ... }
    @Override
    public void rerun() { ... }
    @Override
    protected void reexecuteCommand() { ... }
    protected void determineMigrationFailureForAuditLog() { ... }
    @Override
    protected Guid getCurrentVdsId() { ... }
    public String getDuration() { ... }
    public String getTotalDuration() { ... }
    public String getActualDowntime() { ... }
    @Override
    protected String getLockMessage() { ... }
    private List<Guid> getVdsBlackList() { ... }
    protected List<Guid> getVdsWhiteList() { ... }
    @Override
    public List<PermissionSubject> getPermissionCheckSubjects() { ... }
    @Override
    public void onPowerringUp() { ... }
}
```

AddDiskCommand

```
public class AddDiskCommand<T extends AddDiskParameters> ... {
    protected AddDiskCommand(Guid commandId) { ... }
    public AddDiskCommand(T parameters) { ... }
    public AddDiskCommand(T parameters, CommandContext commandContext) { ... }
    @Override
    protected boolean canDoAction() { ... }
    protected boolean checkIfLunDiskCanBeAdded(DiskValidator diskValidator) { ... }
    protected boolean checkIfImageDiskCanBeAdded(VM vm, DiskValidator diskValidator) { ... }
    private boolean isShareableDiskOnGlusterDomain() { ... }
    private boolean canAddShareableDisk() { ... }
    private boolean checkExceedingMaxBlockDiskSize() { ... }
    private boolean isStoragePoolMatching(VM vm) { ... }
    protected boolean checkImageConfiguration() { ... }
    private double getRequestDiskSpace() { ... }
    @Override
    protected boolean isVmExist() { ... }
    private DiskImage getDiskImageInfo() { ... }
    private boolean isExceedMaxBlockDiskSize() { ... }
    protected DiskLunMapDao getDiskLunMapDao() { ... }
    protected DiskImageDynamicDao getDiskImageDynamicDao() { ... }
    private Guid getDisksStorageDomainId() { ... }
    @Override
    public Guid getStorageDomainId() { ... }
    @Override
    public List<PermissionSubject> getPermissionCheckSubjects() { ... }
    @Override
    protected void setActionMessageParameters() { ... }
    @Override
    protected void executeVmCommand() { ... }
    private void createDiskBasedOnLun() { ... }
    protected VmDevice addManagedDeviceForDisk(Guid diskId, Boolean isUsingScsiReservation) {
        ... }
    protected VmDevice addManagedDeviceForDisk(Guid diskId) { ... }
    protected boolean shouldDiskBePlugged() { ... }
    private void createDiskBasedOnImage() { ... }
    private void createDiskBasedOnCinder() { ... }
    private VdcActionParametersBase buildAddCinderDiskParameters() { ... }
    private void setVmSnapshotIdForDisk(AddImageFromScratchParameters parameters) { ... }
    private void addDiskPermissions(Disk disk) { ... }
    @Override
    public AuditLogType getAuditLogTypeValue() { ... }
    private boolean isDiskStorageTypeRequiresExecuteState() { ... }
    private AuditLogType getExecuteAuditLogTypeValue(boolean successful) { ... }
    protected AuditLogType getEndSuccessAuditLogTypeValue(boolean successful) { ... }
    @Override
    protected VdcActionType getChildActionType() { ... }
    @Override
    protected List<Class<?>> getValidationGroups() { ... }
    @Override
    protected Map<String, Pair<String, String>> getSharedLocks() { ... }
    @Override
    protected Map<String, Pair<String, String>> getExclusiveLocks() { ... }
    @Override
    protected void setLoggingForCommand() { ... }
    private Guid getQuotaId() { ... }
    @Override
    protected void endSuccessfully() { ... }
    private void plugDiskToVmIfNeeded() { ... }
    protected boolean setAndValidateDiskProfiles() { ... }
    @Override
    public List<QuotaConsumptionParameter> getQuotaStorageConsumptionParameters() { ... }
    protected StorageDomainValidator createStorageDomainValidator() { ... }
}
```



Tangled Code in oVirt

CommandBase

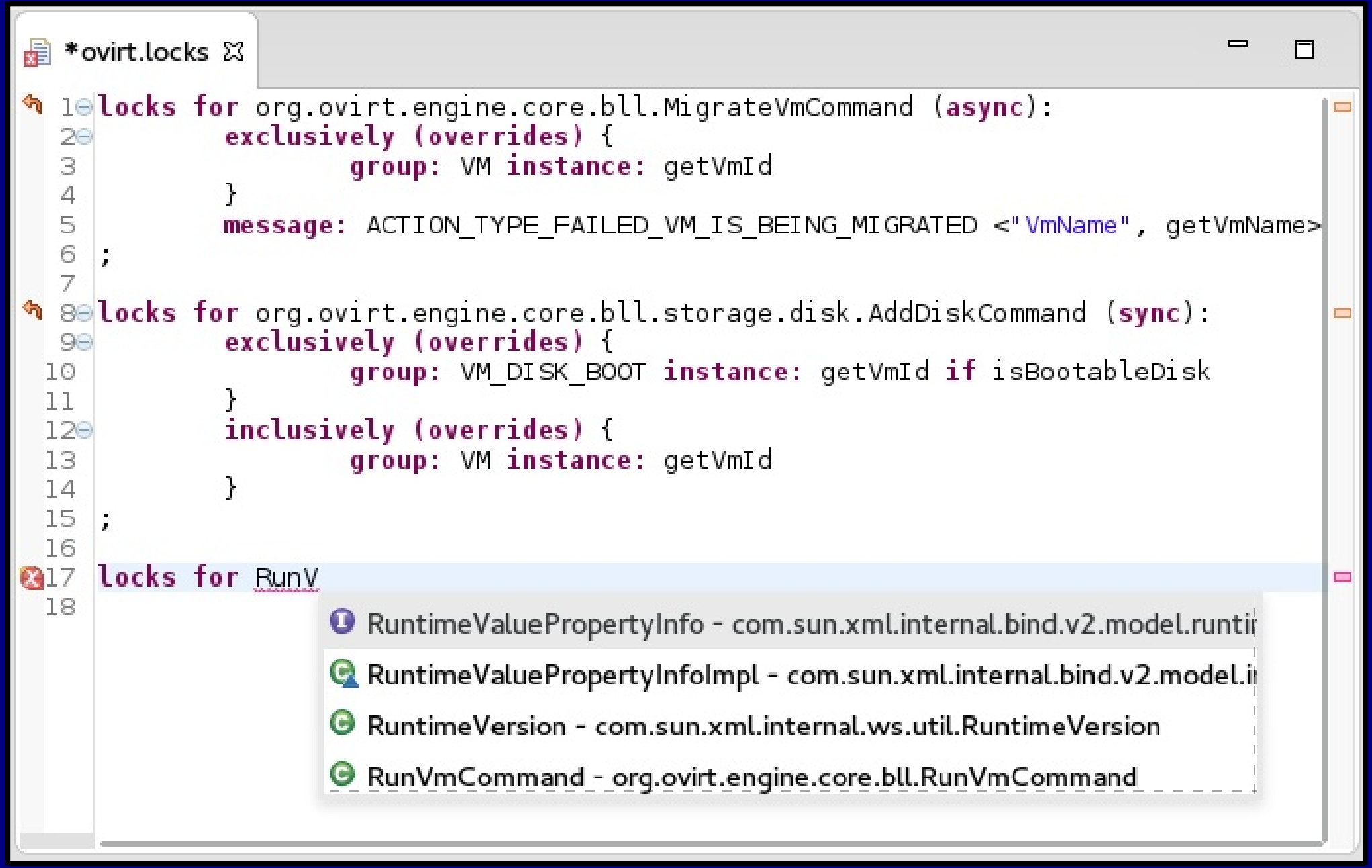
```
private boolean internalCanDoAction() {
 boolean returnValue = false;
 try {
  Transaction transaction = null;
  if (!isCanDoActionSupportsTransaction()) {
   transaction = TransactionSupport.suspend();
  }
  try {
   returnValue =
    isUserAuthorizedToRunAction() && isBackwa      patible()
    && validateInputs() && acquireLock()
    && canDoAction() && internalValidateAndSetQuota();
    if (!returnValue && getReturnValue().getCanDoActionMessages().size() > 0) {
     log.warn("CanDoAction of action '{}' failed for user {}. Reasons: {}",
         getActionType(), getUserName(),
         StringUtils.join(getReturnValue().getCanDoActionMessages(), ','));
    }
  } finally {
   if (transaction != null) {
    TransactionSupport.resume(transaction);
   }
  }
 } catch (DataAccessException dataAccessEx) {
  log.error("Data access error during CanDoActionFailure.", dataAccessEx);
  addCanDoActionMessage(EngineMessage.CAN_DO_ACTION_DATABASE_CONNECTION_FAILURE);
 } catch (RuntimeException ex) {
  log.error("Error during CanDoActionFailure.", ex);
  addCanDoActionMessage(EngineMessage.CAN_DO_ACTION_GENERAL_FAILURE);
 } finally {
  if (!returnValue) {
   freeLock();
  }
 }
 return returnValue;
}
```

Synchronization

Permissions

Synchronization

Example: Using ovirtSync



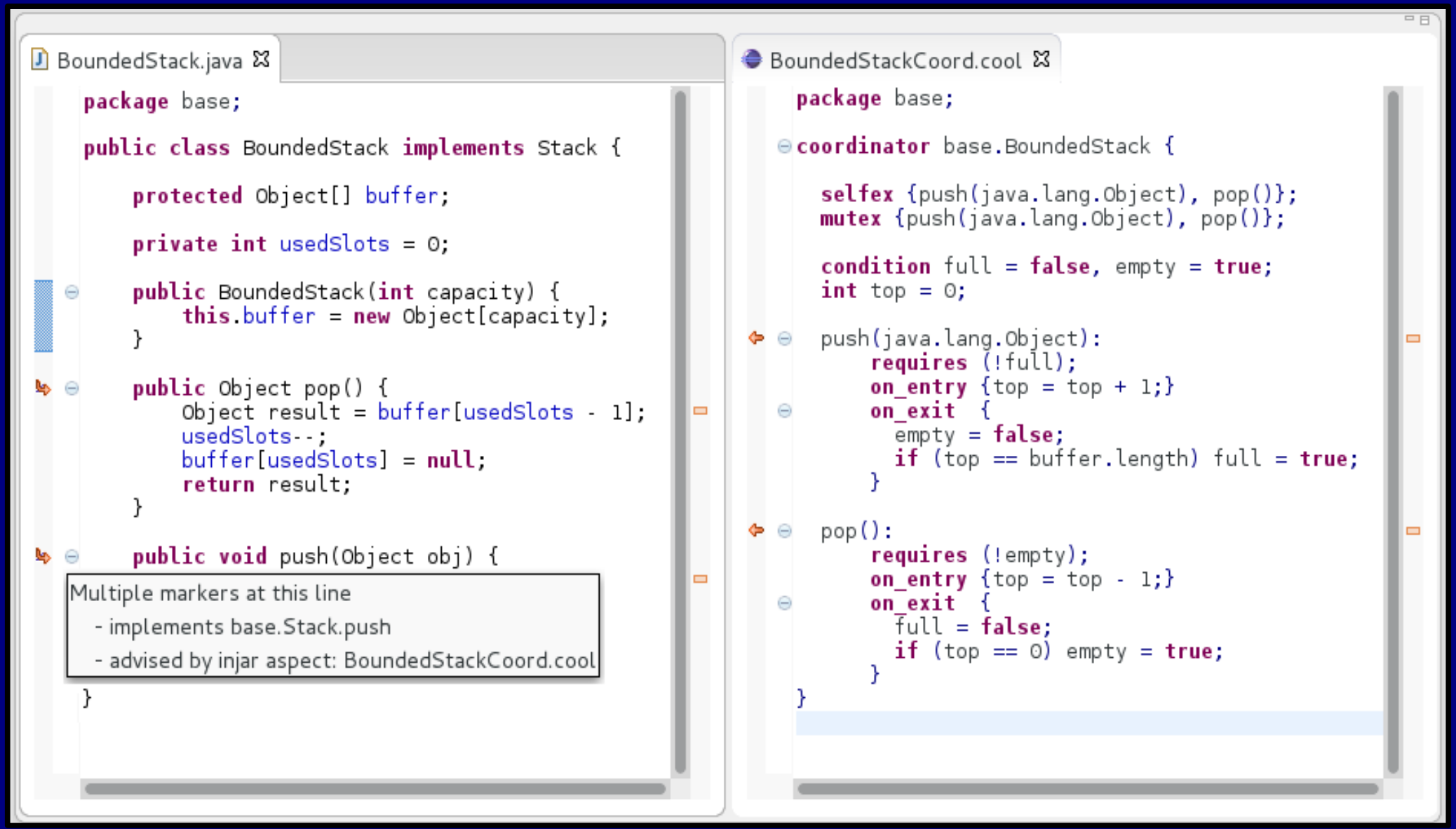
Lessons from Case Study II

- **Separated out crosscutting concerns**
  - Scattered code (over 25% of some classes) is encapsulated in a separate module
  - Tangled code is extracted from the root class (over 12% from its LOC)
- **Practical language development**
  - Few hours per DSAL with supportive tools
- **Effective programming with multiple DSALs simultaneously**
  - In a large-scale and complex project

Implementing COOL

- **Language implementation**
  - Implementation of a complex third-party DSAL for thread synchronization
- **Language use**
  - Implementation of an aspect solution for the bounded-stack benchmark example
- **Baseline for comparison**
  - The implementation of COOL in the AWESOME composition framework

Synchronizing a Bounded-Stack



BoundedStack.java

```
package base;

public class BoundedStack implements Stack {

    protected Object[] buffer;

    private int usedSlots = 0;

    public BoundedStack(int capacity) {
        this.buffer = new Object[capacity];
    }

    public Object pop() {
        Object result = buffer[usedSlots - 1];
        usedSlots--;
        buffer[usedSlots] = null;
        return result;
    }

    public void push(Object obj) {
```

Multiple markers at this line
- implements base.Stack.push
- advised by injar aspect: BoundedStackCoord.cool

```
}
```

BoundedStackCoord.cool

```
package base;

coordinator base.BoundedStack {

  selfex {push(java.lang.Object), pop()};
  mutex {push(java.lang.Object), pop()};

  condition full = false, empty = true;
  int top = 0;

  push(java.lang.Object):
      requires (!full);
      on_entry {top = top + 1;}
      on_exit  {
        empty = false;
        if (top == buffer.length) full = true;
      }

  pop():
      requires (!empty);
      on_entry {top = top - 1;}
      on_exit  {
        full = false;
        if (top == 0) empty = true;
      }
}
```

Avoid Multi-DSAL Conflicts

- Without @Hide
  - Known multi-DSAL conflicts reproduced
- With @Hide
  - Multi-DSAL conflicts not observed

```
public aspect AJAuditor {
    pointcut toLog(): call(* *.*(..)) &&
                      !cflow(within(AJAuditor));
    before(): toLog() { log(thisJoinPoint); }
    protected void log(JoinPoint jp) {
        BoundedStack buf = BoundedStack.getInstance();
        try { if (buf != null) buf.add(jp); }
        catch(Exception e) { /** ... skipped ... */ }
    }
}
```

Implementation Effort

Implementation	Grammar	Code Transformation		Weaver Plugin
		EV	Other	
Language	SDF	Stratego (AST)	Stratego (AST)	Java
CF Approach	34	761 (4168)	297 (3001)	1557
Our Approach	34	0	382 (3008)	0

- Our implementation vs. the alternative
  - Significantly less code required
  - More high-level
    - AspectJ vs bytecode manipulation
  - Language Workbench compatible
    - Done completely in Spoofax

Related Work

- **Domain Specific Aspect Languages**
  - [Fabry at al., 2015] A Taxonomy of Domain-Specific Aspect Languages.
- **Transformation-based AOP Composition Frameworks**
  - [Shonle at al., 2003] XAspects: An extensible system for domain specific aspect languages.
  - [Tanter, 2006] Aspects of composition in the Reflex AOP kernel.
- **The AWESOME Composition Framework**
  - [Kojarski and Lorenz, 2007] Identifying feature interaction in aspect-oriented frameworks.
  - [Kojarski and Lorenz, 2007] Awesome: An aspect co-weaving system for composing multiple aspect-oriented extensions.
- **SpecTackle**
  - [Lorenz and Mishali, 2012] SpecTackle: Toward a specification based DSAL composition process.

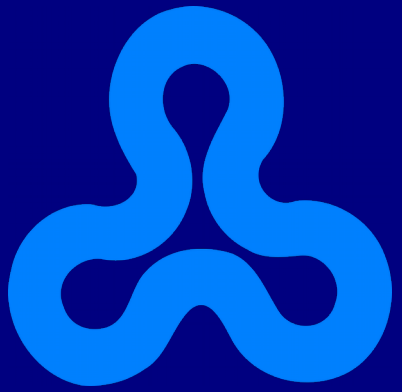
Conclusion

- **LOM can follow a similar process to LOP**
  - For a class of DSALs that are in a sense reducible to a GPAL
- **DSALs become cost-effective**
  - The implementation cost is reduced
  - The effectiveness of using them is increased
- **LOM becomes practical for real-world software development process**

The code is available on GitHub:

https://github.com/OpenUniversity/

Thank You!



Arik Hadas and **David H. Lorenz**
Dept. of Mathematics and Computer Science
The Open University of Israel

arik.hadas@openu.ac.il

https://github.com/OpenUniversity