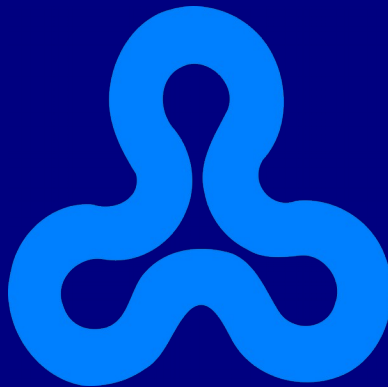


Toward Disposable Domain-Specific Aspect Languages

Arik Hadas

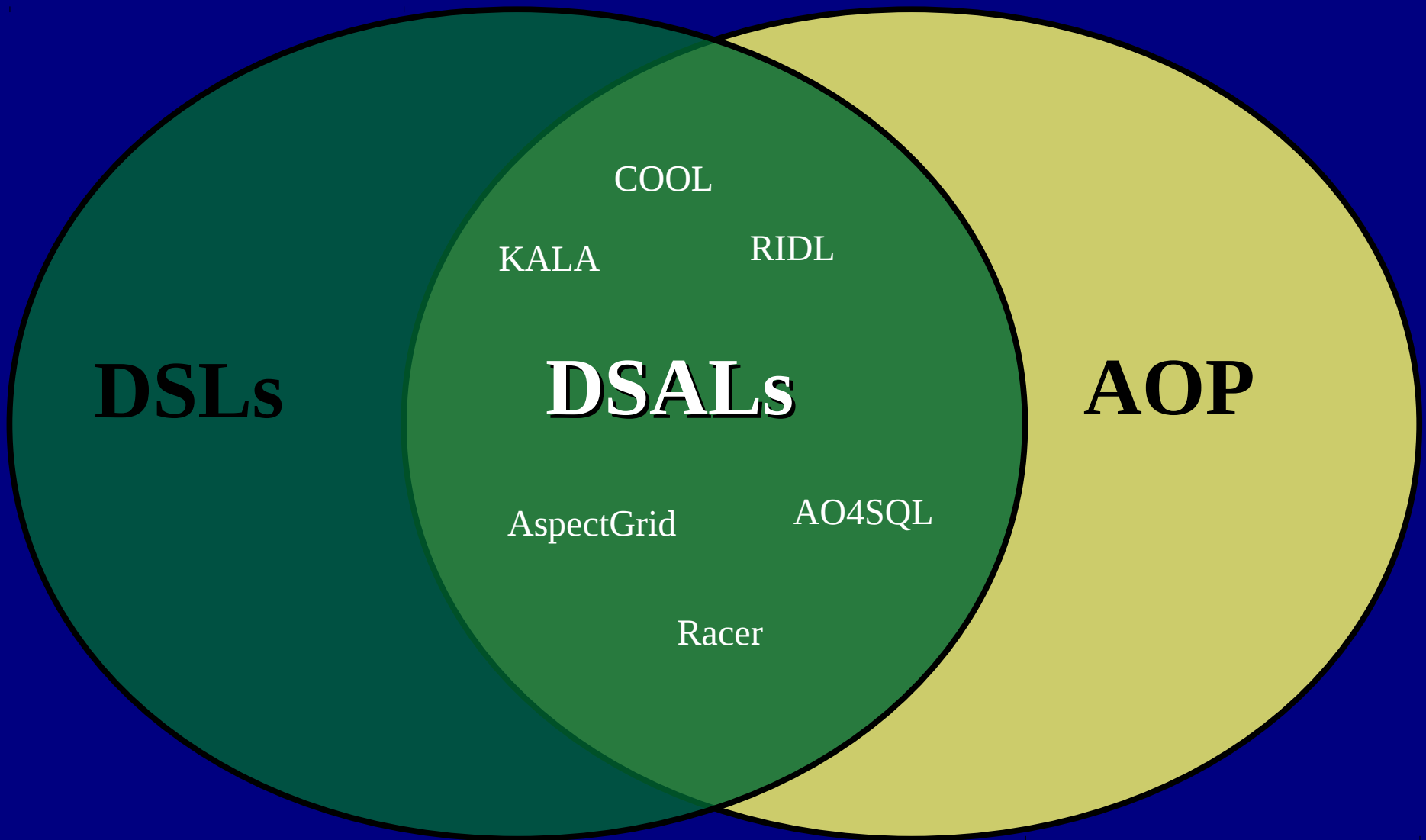
Dept. of Mathematics and Computer Science
The Open University of Israel



Joint Work With:

David H. Lorenz

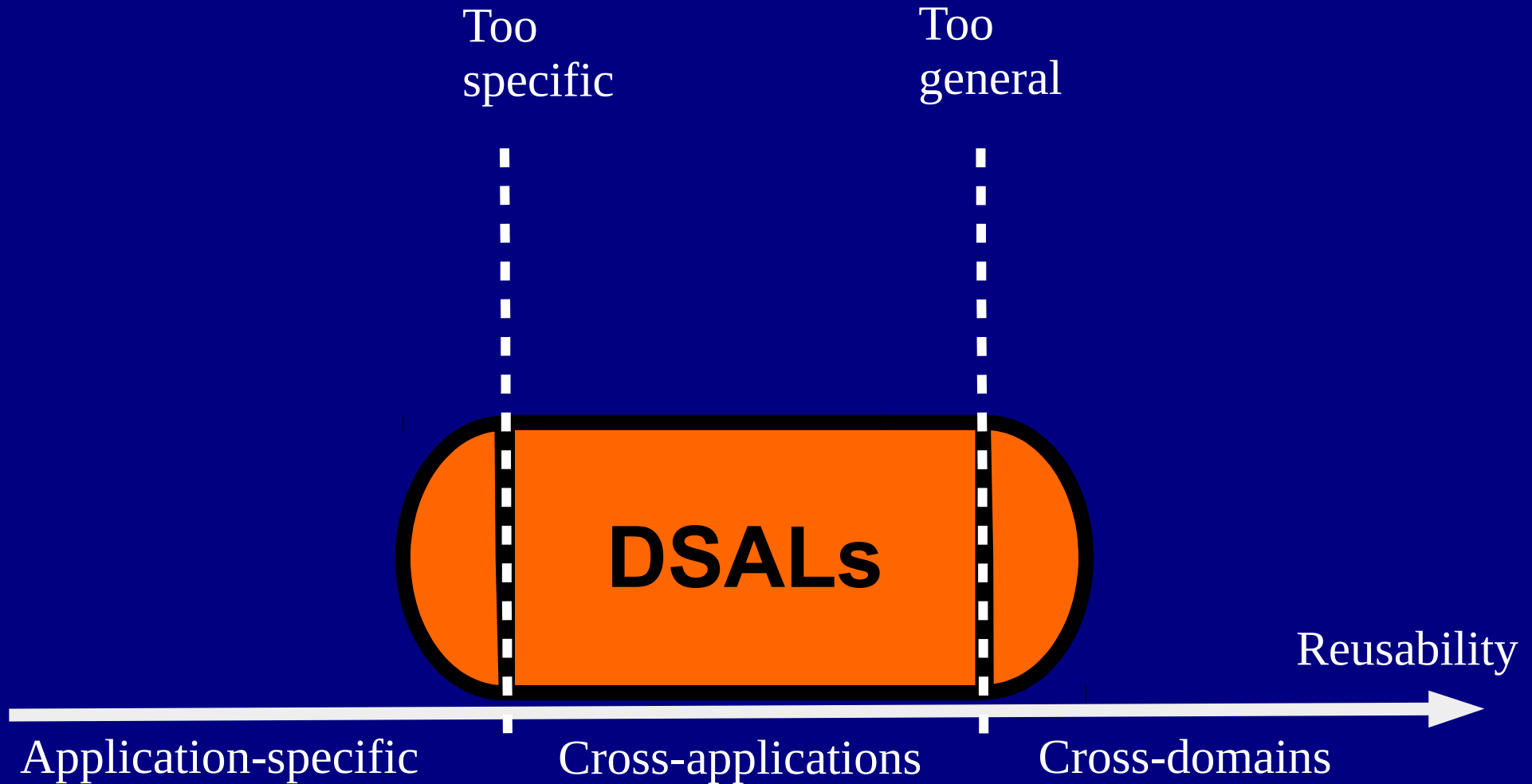
Domain Specific Aspect Languages



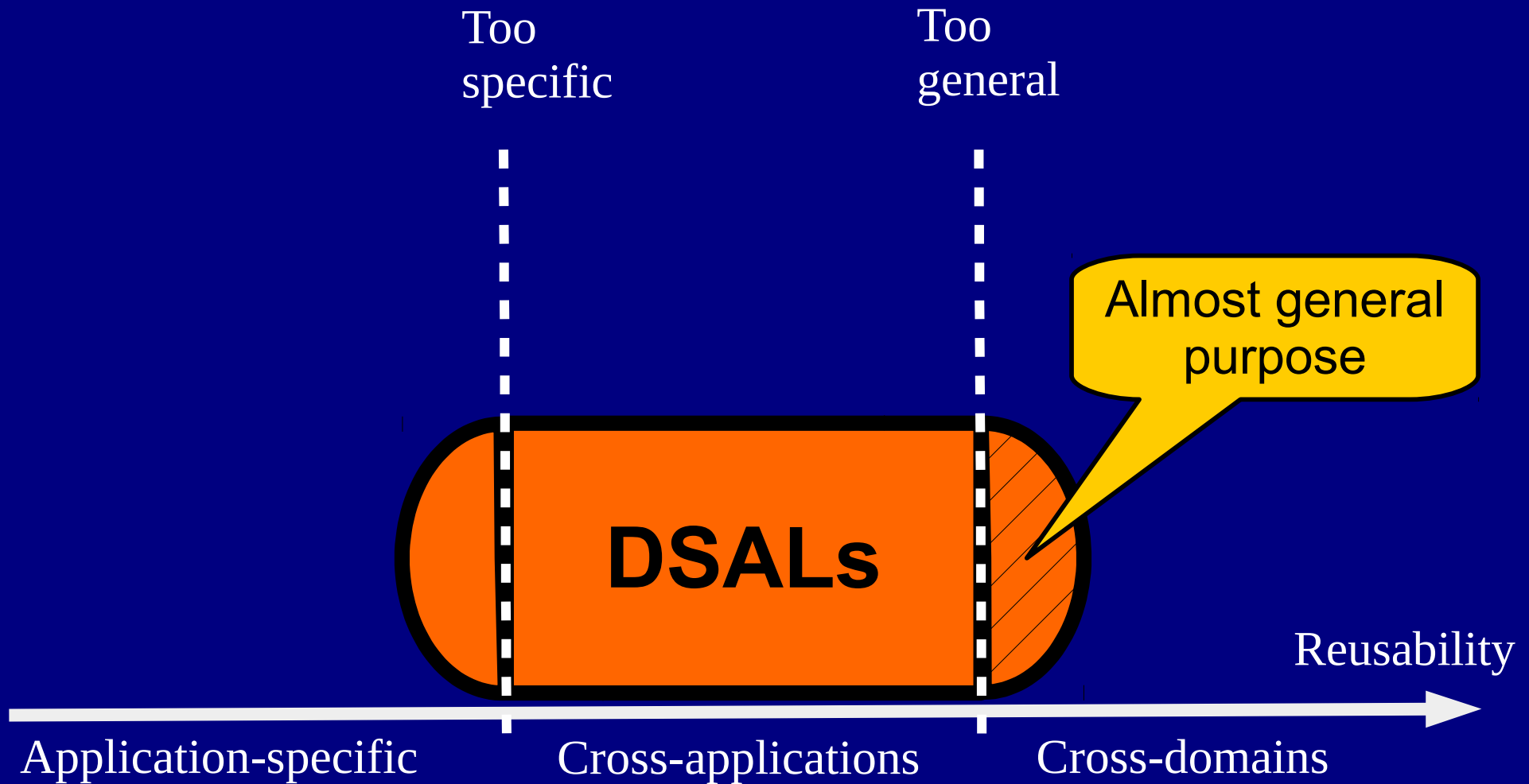
The DSAL Spectrum



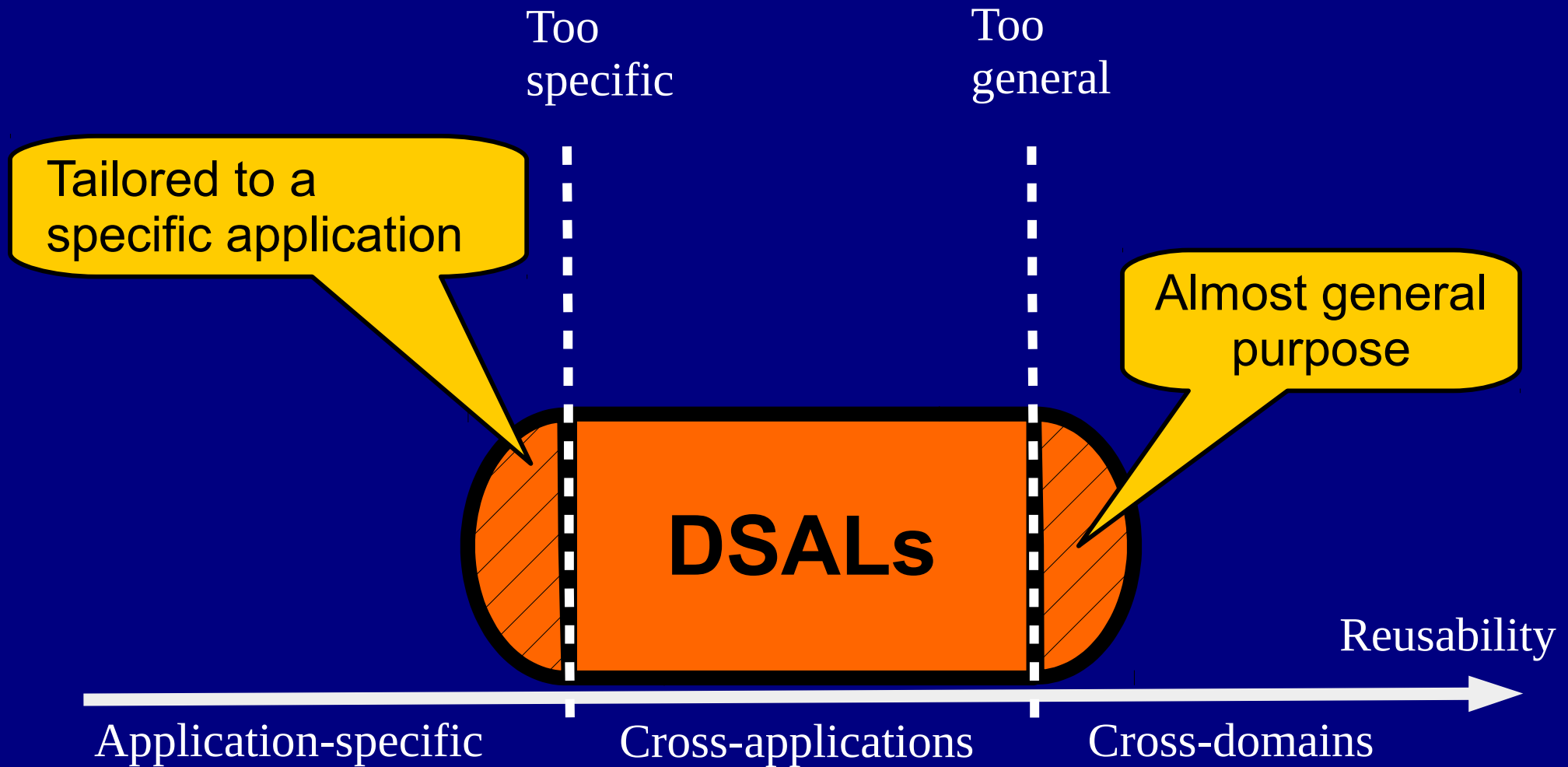
The DSAL Spectrum



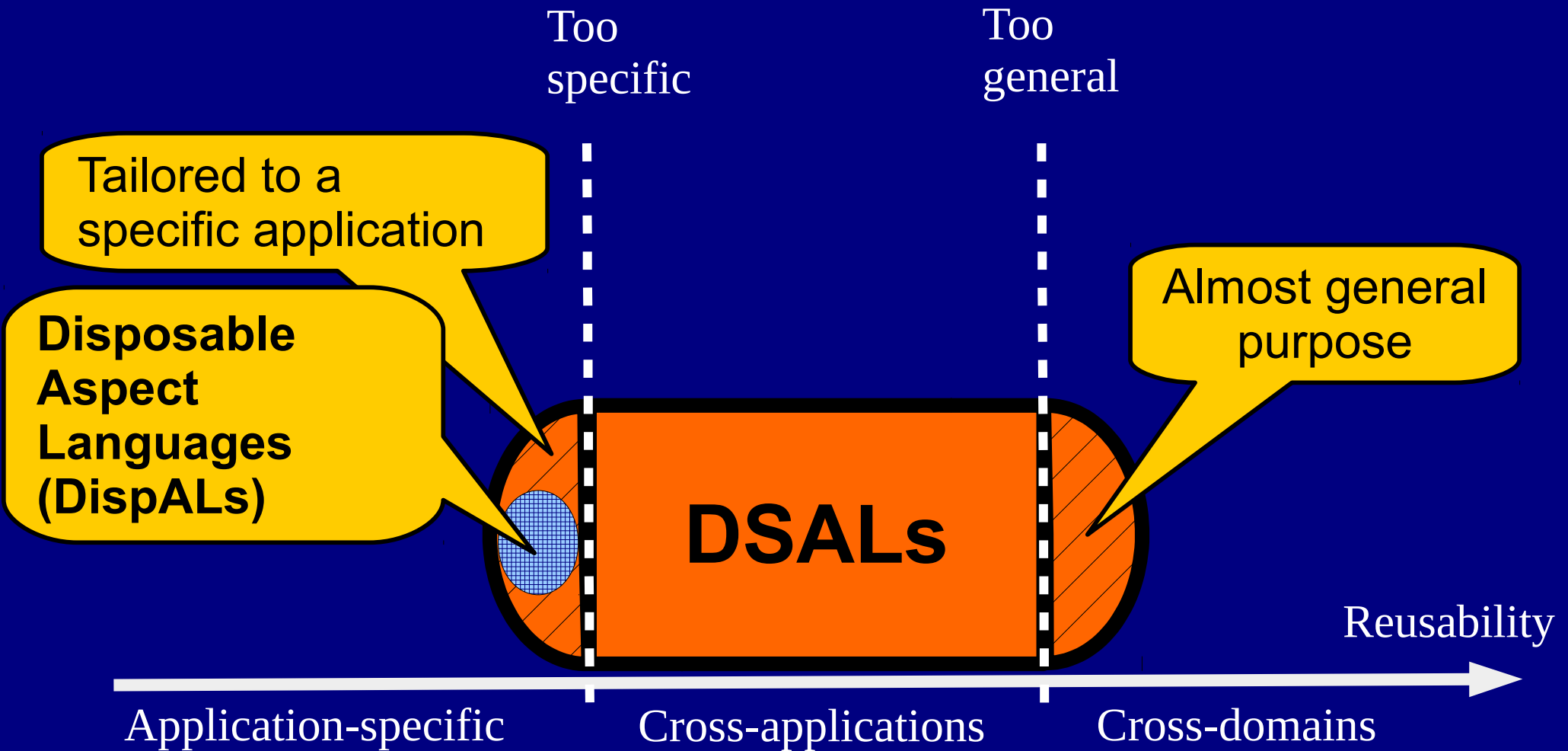
The DSAL Spectrum



The DSAL Spectrum



The DSAL Spectrum



DispALs – (Not) a Crazy Idea

- **Disposable Aspect Languages (DispALs)**
 - DSALs used once and thrown away
- **What makes DispALs practical**
 - Modern tools significantly reduce their implement cost
 - The reduced implementation cost leads to simpler DispALs
 - Reduce their definition and implementation even further

Outline

- Introduction
- **Motivation**
- Approach
- Conclusion

Simple Auditing with AspectJ

```
public aspect SimpleAspect {  
  
    void around(Command command):  
        execution(* execute()) && this(command) {  
        try {  
            proceed(command);  
        }  
        finally {  
            if (command instanceof CopyCommand<?>) {  
                audit(command.isSucceeded() ?  
                    "copy has been started" : "copy failed");  
            }  
        }  
    }  
  
    ...  
  
}
```

Additional
constructs

Starts
simple

Enhanced Auditing with AspectJ

```
public privileged aspect EnhancedAspect {
    void around(Command command): execution(* execute()) && this(command) {
        try {
            proceed(command);
        }
        finally {
            if (command instanceof CopyCommand<?>) {
                CopyCommand<?> copyCmd = (CopyCommand<?>) command;
                CopyParameters params = copyCmd.getParameters();
                if (!command.isSucceeded()) {
                    audit(resolve(AuditMessages.COPY_FAILED, params.getResource()));
                } else {
                    if (command.isAsync()) {
                        String msg = resolve(
                            copyCmd.encrypt() ?
                                AuditMessages.COPY_ENCRYPT_STARTED : AuditMessages.COPY_STARTED,
                            params.getResource(), params.getSource(), params.getDestination());
                        audit(msg);
                    } else {
                        audit(resolve(AuditMessages.COPY_SUCCEEDED, params.getResource(),
                            params.getSource(), params.getDestination()));
                    }
                }
            }
        }
    }
    ...
    ...
}
```

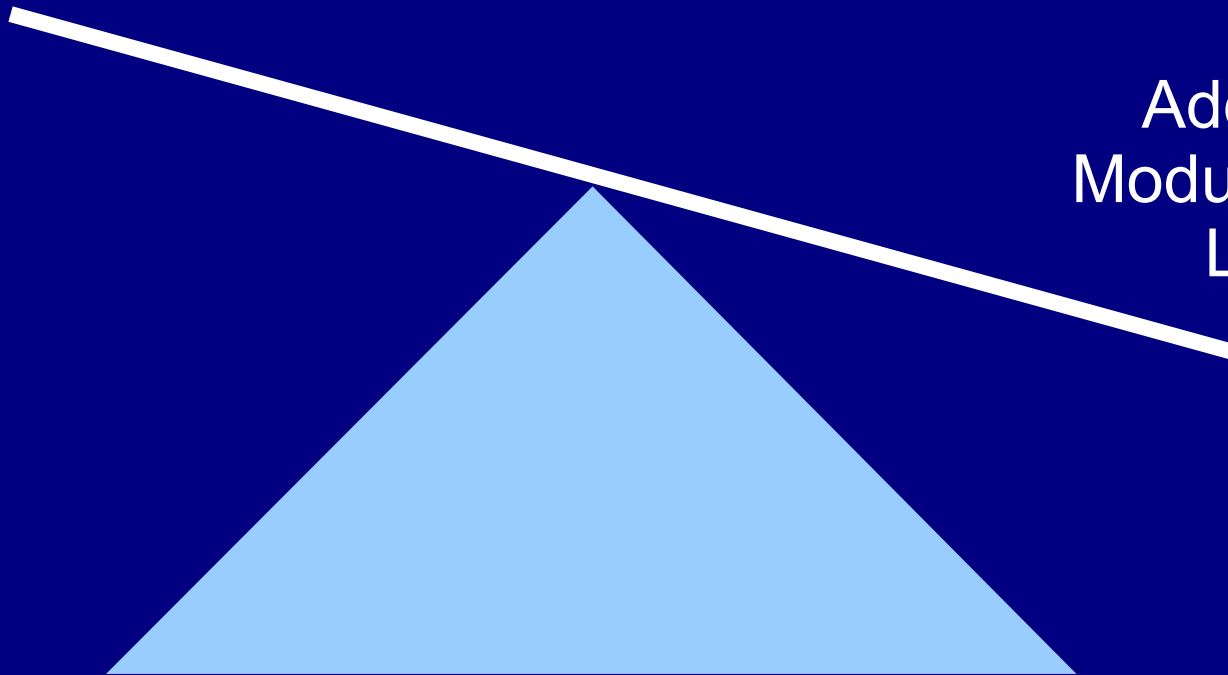


Quickly
becomes
complex

AspectJ Trade-offs

Complexity of
Programming
Language

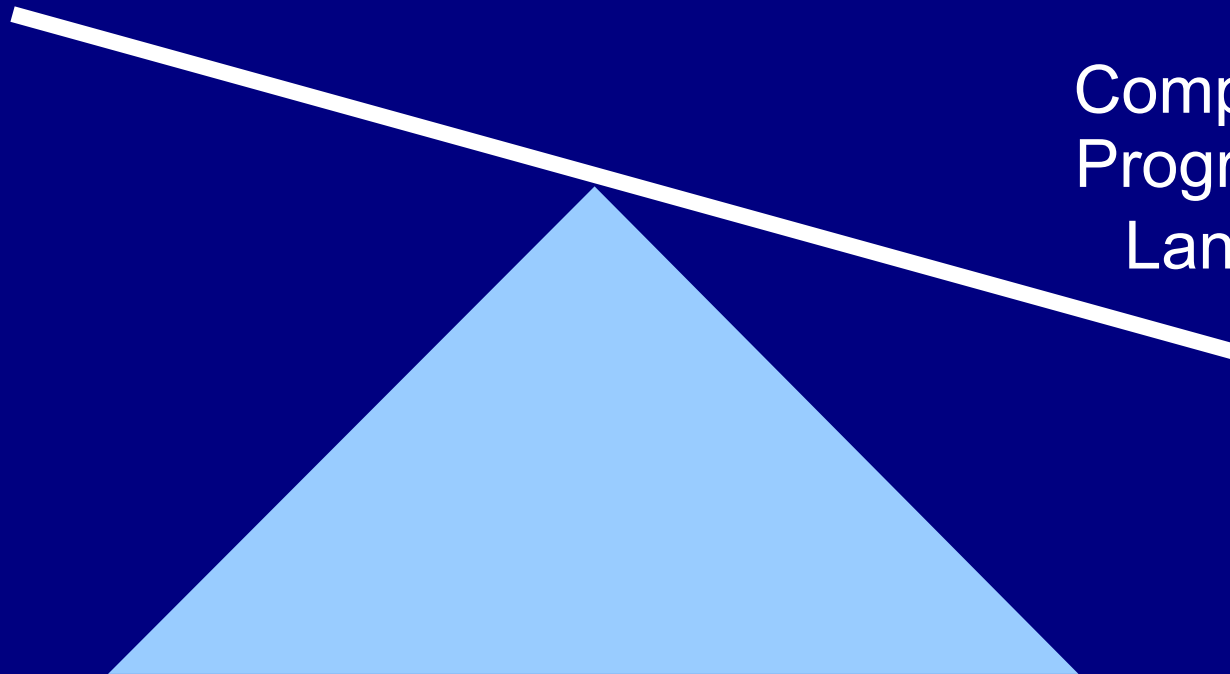
Additional
Modularization
Layer



DSAL Trade-offs

Cost-effectiveness of
Development and Use

Complexity of
Programming
Language



DispAL Balance the Trade-Offs











- **Improves software modularity**
 - Separation of crosscutting concerns
- **Reduces the complexity of the language**
 - Domain-specific
- **More cost-effective**

Enhanced Auditing with a DispAL

```
logs for demo.CopyCommand:  
  case failure  
    log (COPY_FAILED)  
  case started & encrypt  
    log (COPY_ENCRYPT_STARTED, getResource, getSource, getDestination)  
  case started  
    log (COPY_STARTED, getResource, getSource, getDestination)  
  case success  
    log (COPY_SUCCEEDED, getResource, getSource, getDestination)  
;
```

Configuration
like

Language Comparison

| | AspectJ | Ordinary DSAL | Disposable DSAL |
|-------------------------|---|---|---|
| Language Reuse |  |  |  |
| Language Design | |  |  |
| Language Implementation | |  |  |
| Language Use |  |  |  |

Outline

- Introduction
- Motivation
- **Approach**
- Conclusion

Transformation-based Approach

- **Transform DispALs to GPAL-based Kernel**
 - No need to implement a dedicated weaver
 - Can leverage language workbenches
 - Can leverage GPAL development tools
- **Reuse the compiler**
 - One time effort to develop the compiler

Evaluation

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

Scattered Code in oVirt-Engine

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 initVds() { ... }
    private List<Guid> getDestinationHostList() { ... }
    @Override
    protected void executeVmCommand() { ... }
    private boolean perform() { ... }
    private boolean migrateVm() { ... }
    private MigrateVDSCommandParameters createMigrateVDSCommandParameters() { ... }
    @Override
    public void runningSucceeded() { ... }
    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 onPoweringUp() { ... }
}
```

synchronization

Auditing

Permissions

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-Engine

- The code in the common root of all commands called **CommandBase** is tangled

```
private boolean internalCanDoAction() {
    boolean returnValue = false;
    try {
        Transaction transaction = null;
        if (!isCanDoActionSupportsTransaction()) {
            transaction = TransactionSupport.suspend();
        }
        try {
            returnValue =
                isUserAuthorizedToRunAction() && isBackwardsCompatible()
                && validateInputs() && acquireLock()
                && canDoAction() && internalValidateAndSetQuote();
            if (!returnValue && getReturnValue().getCanDoActionMessages().size() > 0) {
                log.warn("CanDoAction of action '{0}' failed for user {1}. Reasons: {2}",
                    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;
}
```

permissions

synchronization

synchronization

Implementation Effort

- **One time effort**
 - Compiler for the kernel language
- **Per-application effort**
 - Compile oVirt with AspectJ compiler
- **The produced DispALs were**
 - Relatively easy to define
 - Relatively easy to implement
 - Relatively easy to use

Grammar Definition of ovirt-auditing

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

```
Command:
```

```
'logs for' type=[types::JvmDeclaredType|QualifiedName]
  (overrides?='(overrides)')? ':'
    (cases+=Case(',', cases+=Case)* (',' 'otherwise' 'log'
    default=[types::JvmEnumerationLiteral]))?)?
';'
;
```

```
Case:
```

```
'case' (actionState=[types::JvmEnumerationLiteral] '&')?
  result=Result('&' internal?='internal')?('&'
  'state='(fields+=[types::JvmField]))*('&'
  (methods+=[types::JvmOperation]))* 'log'
  msg=[types::JvmEnumerationLiteral]
;
```

```
enum Result:
```

```
  success|failure
```

```
;
```

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

Transformation of ovirt-auditing

```
override void doGenerate(Resource resource, IFileSystemAccess fsa) {  
    var path = 'org.ovirt.engine.core.bll.'.replaceAll('\\.', File.separator) + 'Logs.aj'  
    fsa.generateFile(path, resource.compile)  
}
```

```
def compile(Resource resource) {  
    this.resource = resource  
    ...  
    package org.ovirt.engine.core.bll;  
  
    import org.aspectj.lang.annotation.BridgedSourceLocation;  
    import org.ovirt.engine.core.common.AuditLogType;  
    import org.ovirt.engine.core.bll.CommandActionState;  
  
    public privileged aspect Logs {  
        «FOR command:resource.allContents.filter(typeof(Command)).toIterable»  
        «command.compile»  
        «ENDFOR»  
    }  
    ...  
}
```

```
def compile(Command command) '''  
    «NodeModelUtils.getNode(command).toSourcePosition»  
    AuditLogType around(«command.type.qualifiedName» command): execution(*  
getAuditLogTypeValue()) && this(command) {  
        «FOR acase:command.cases»  
        «acase.compile»  
        «ENDFOR»  
        return «IF command.^default != null»AuditLogType.«command.^default.simpleName»«ELSEIF  
command.overrides»AuditLogType.UNASSIGNED«ELSE»proceed(command)«ENDIF»;  
    }  
    ...
```

... skipped...

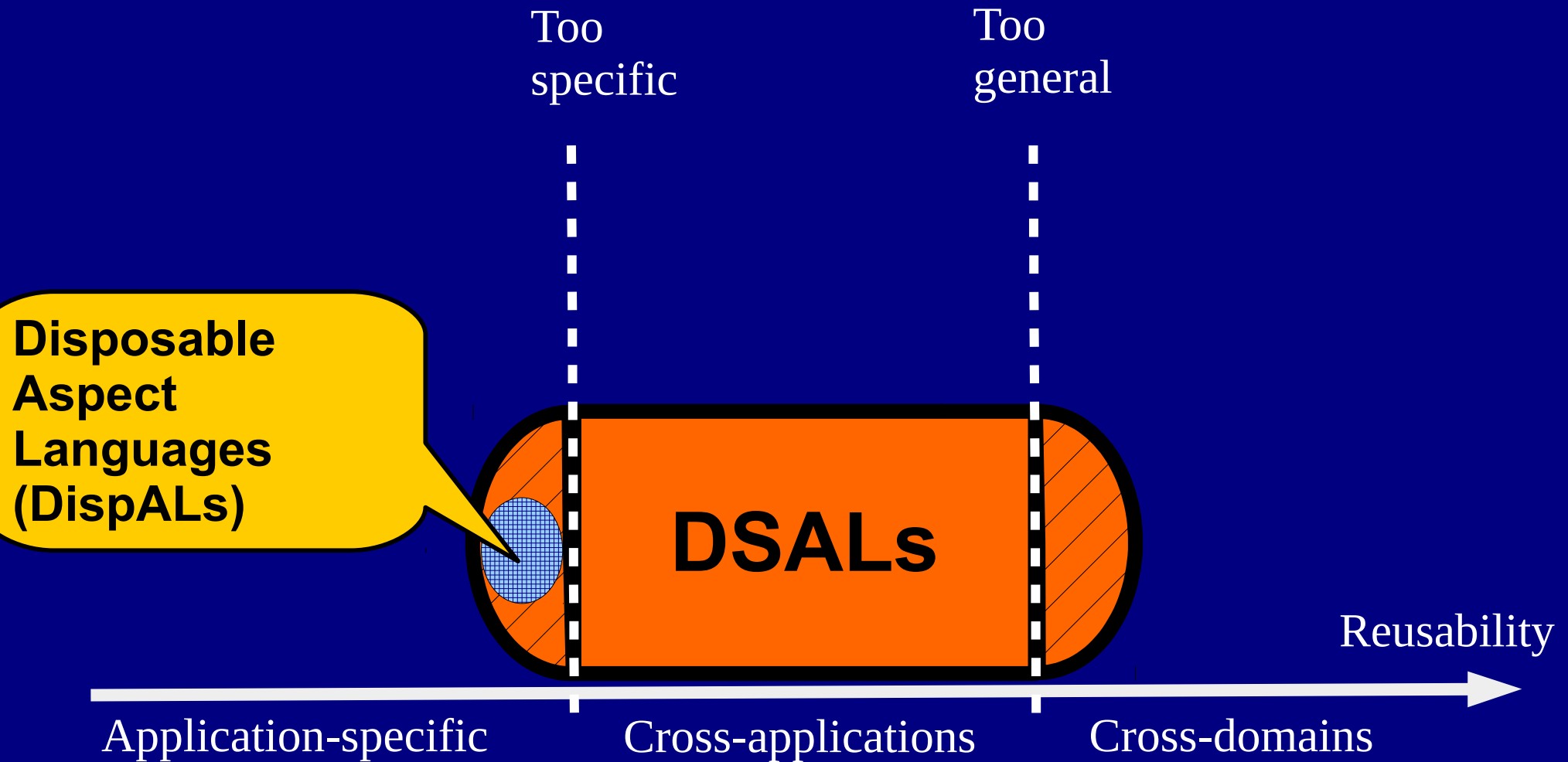
Outline

- Introduction
- Motivation
- Approach
- **Conclusion**

Related Work

- **Domain specific aspect languages**
 - **[Lopes and Kiczales, 1998]** D: A language framework for distributed computing.
- **Language Oriented Modularity**
 - **[Lorenz, 2012]** Language-oriented modularity through Awesome DSALs: summary of invited talk.
- **Making LOM practical**
 - **[Hadas and Lorenz, 2015]** Demanding first-class equality for domain specific aspect languages.

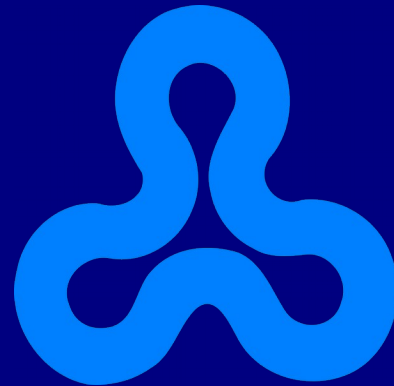
The DSAL Spectrum



Summary

- **Even disposable DSALs may be cost-effective**
 - For CCC that are modularizable using a GPAL
 - Leveraging a language workbench
- **DispALs are preferable to ordinary DSALs or GPALs**
 - For CCC that are
 - Complex to express in GPALs
 - Simple to express in domain-specific syntax
 - Highly coupled with the business logic

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>