# Application-Specific Language-Oriented Modularity: A Case Study of the oVirt Project

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

Joint Work With:
**David H. Lorenz**

# Application-Specific LOM

- Highly specific DSALs
  - Designed to the problem at hand
- Low reusability
  - No expectation for reuse across applications
  - Reusable between application-versions

# oVirt – Open Virtualization

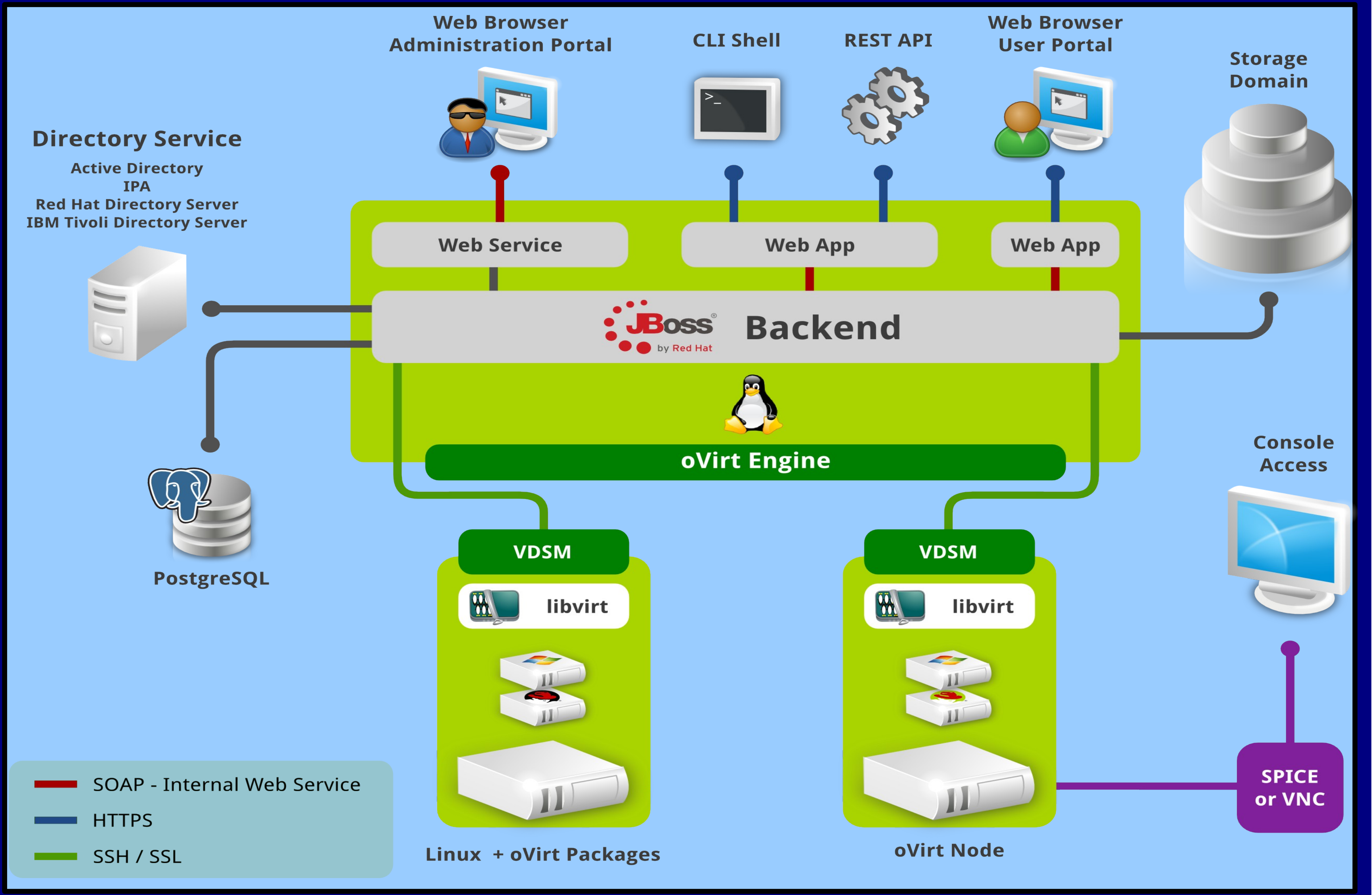
- **oVirt: enterprise application for providing and managing virtual data centers**
  - Open-source
  - Manages various aspects of running virtual machines on top of the KVM hypervisor
    - Network, Storage, SLA, etc.
  - The upstream of Red Hat Enterprise Virtualization
  - Alternative to VMware's vSphere

# oVirt in Universidad de Sevilla

When one of the largest universities in **Spain** needed a virtualization solution to host their virtual desktop interface program, UDS Enterprise helped the institution find a virtualization solution that delivered superior flexibility at a much lower cost than proprietary solutions.

That solution would be oVirt. Today, more than 3,000 students use this virtual desktop infrastructure, with the prospect of the rest of the student body participating as the program grows.

# oVirt's Architecture

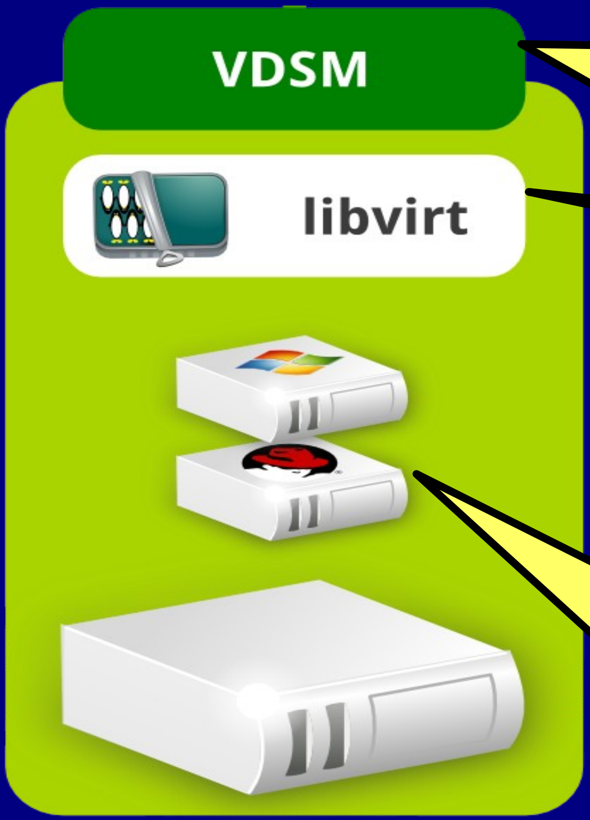


# oVirt-Engine

- oVirt-Engine: the control center of oVirt
  - Java server application (on top of JBoss)
  - Executes operations it gets from clients
  - Reports the up-to-date status of the data center
  - Consists on 761K lines-of-code, contributed by 173 developers
  - Its design is based on the COMMAND design pattern
    - Easier to understand
    - Allow to treat the commands uniformly

# oVirt – Case in Study

- **Multi-layer architecture on the hosts leads to applications with limited responsibilities**



VDSM is responsible for the communication with oVirt-Engine and for host-networks and storage

Libvirt provides simpler API for clients with various language bindings

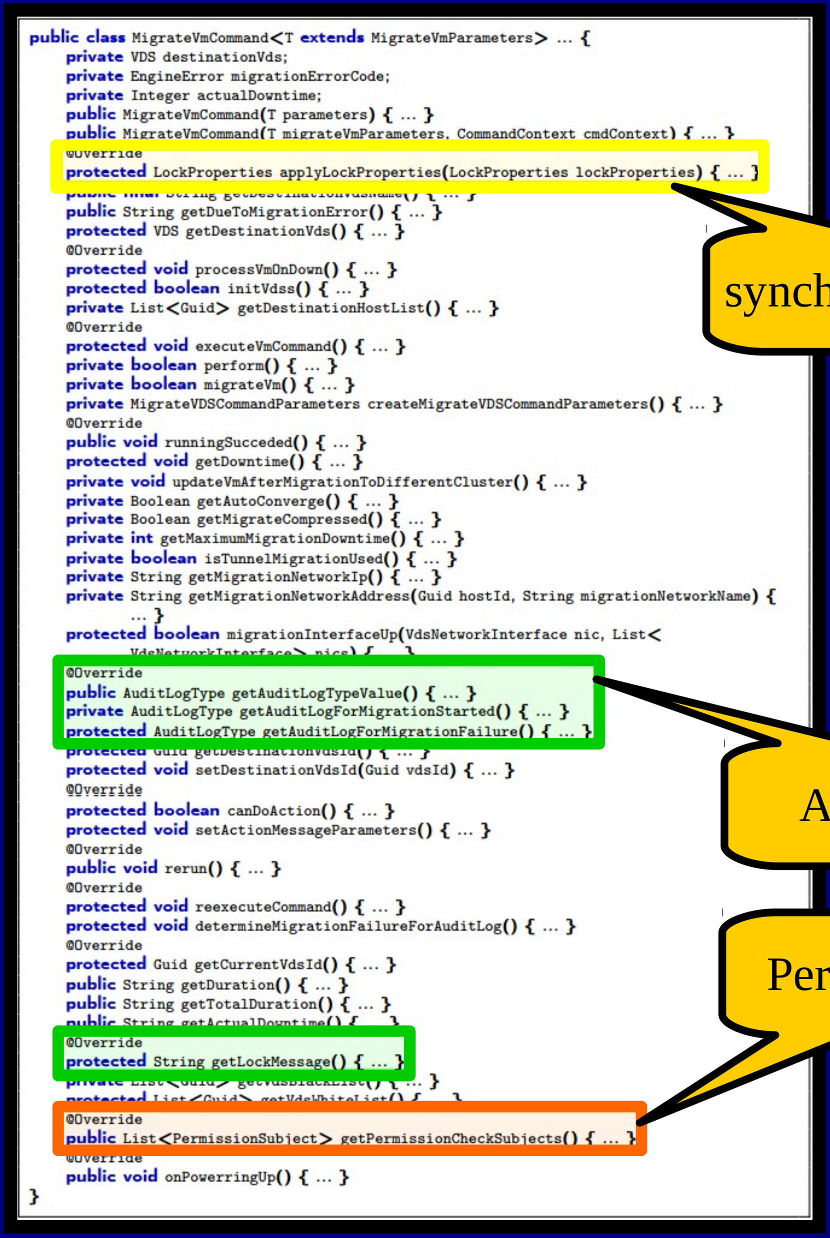
Hypervisor is responsible for low-level things required for the emulation of a virtual environment, like memory management, driver emulations, etc.

# oVirt – Case in Study

- Multi-layer architecture on the hosts lead to applications with limited responsibilities
- However, oVirt-Engine is complex
  - Many responsibilities
  - Lot of code
- We found several crosscutting concerns in oVirt-Engine:
  - Synchronization
  - Auditing
  - Permission checks

# Scattered Code in oVirt-Engine

## MigrateVmCommand



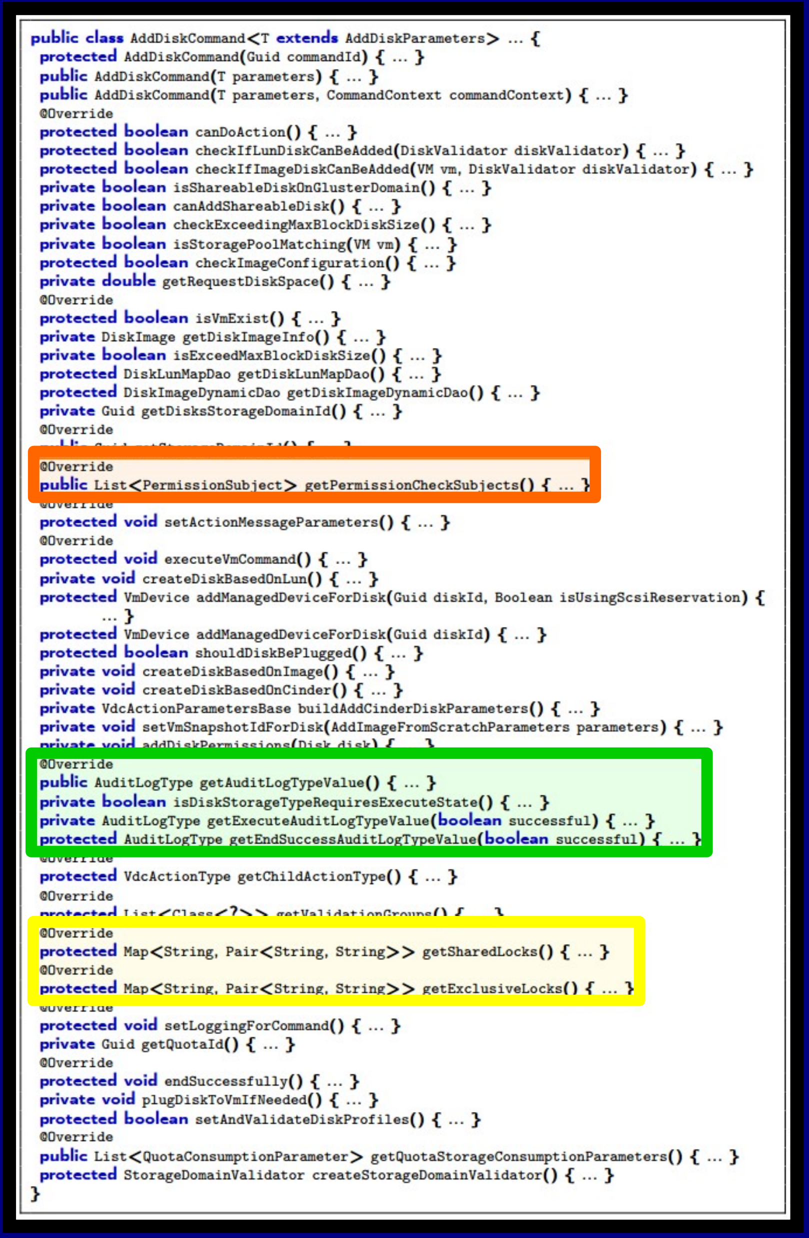
```
public class MigrateVmCommand<T extends MigrateVmParameters> ... {
    private VDS destinationVds;
    private EngineError migrationErrorCode;
    private Integer actualDowntime;
    public MigrateVmCommand(T parameters) { ... }
    public MigrateVmCommand(T migrateVmParameters, CommandContext cmdContext) { ... }
    @Override
    protected LockProperties applyLockProperties(LockProperties lockProperties) { ... }
    public final String getDestinationVdsName() { ... }
    public String getDueToMigrationError() { ... }
    protected VDS getDestinationVds() { ... }
    @Override
    protected void processVmOnDown() { ... }
    protected boolean 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() { ... }
}
```

synchronization

Auditing

Permissions

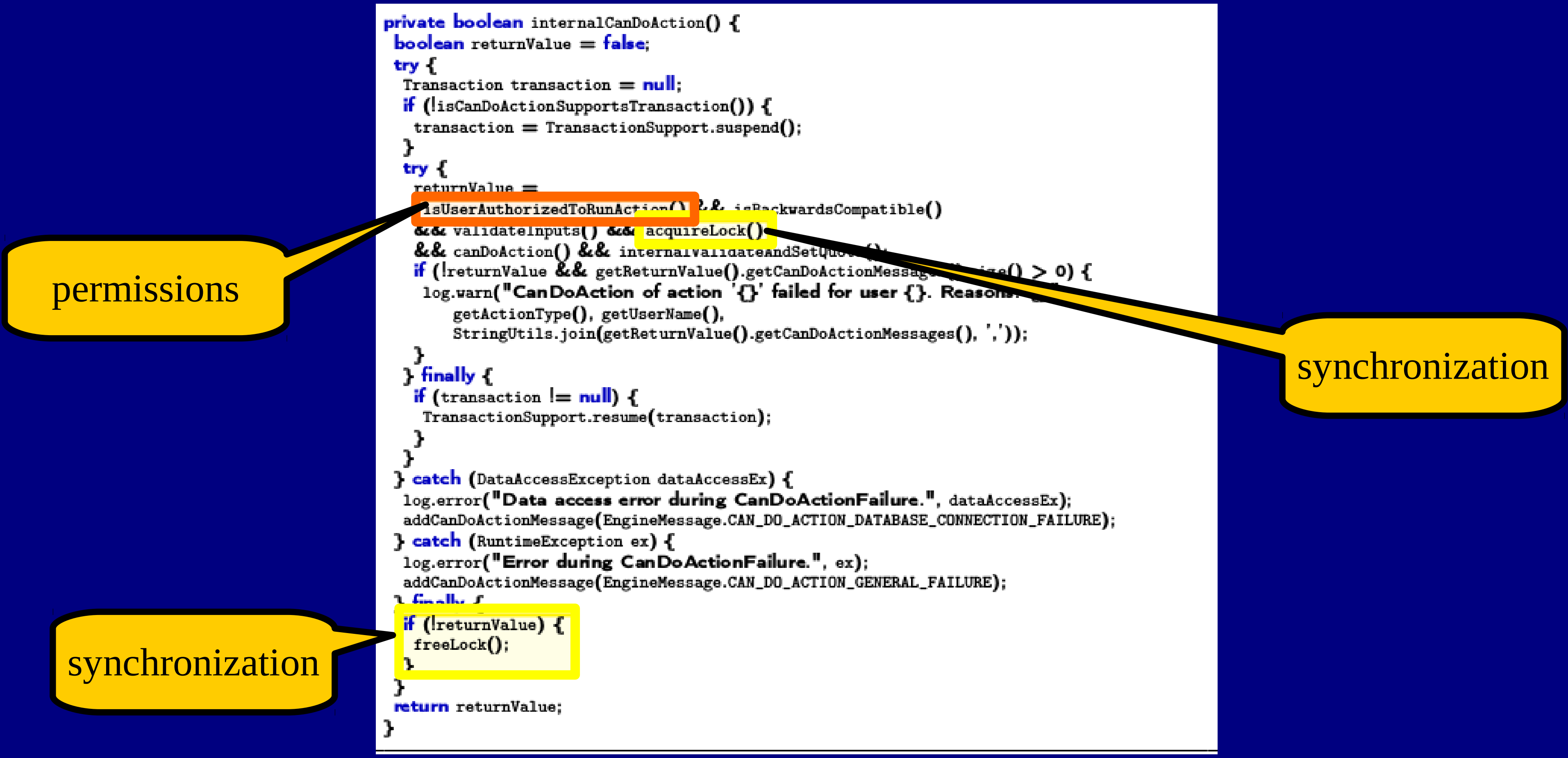
## AddDiskCommand



```
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

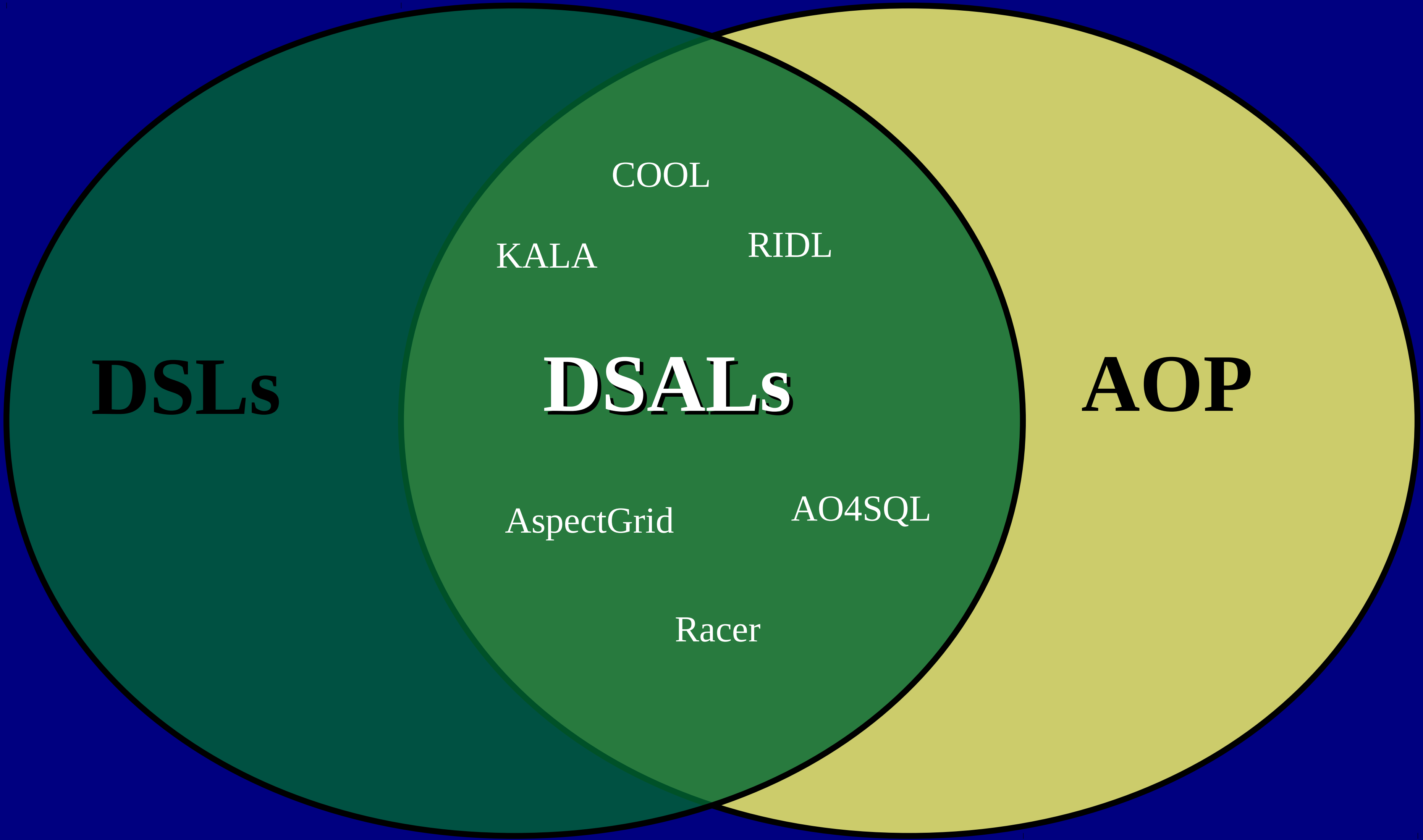


```
private boolean internalCanDoAction() {
 boolean returnValue = false;
 try {
  Transaction transaction = null;
  if (!isCanDoActionSupportsTransaction()) {
   transaction = TransactionSupport.suspend();
  }
  try {
   returnValue =
   isUserAuthorizedToRunAction() && isBackwardsCompatible()
   && validateInputs() && acquireLock()
   && canDoAction() && 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;
}
```

# Outline

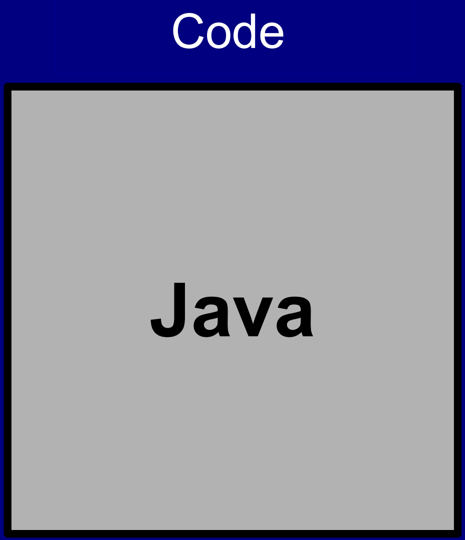
- Introduction
- Application Specific LOM
- Demonstration
- Conclusion

# Domain Specific Aspect Languages



# Language Oriented Modularity (LOM)

- **A methodology that puts Domain Specific Aspect Languages (DSALs) at the center of the software modularization process.**

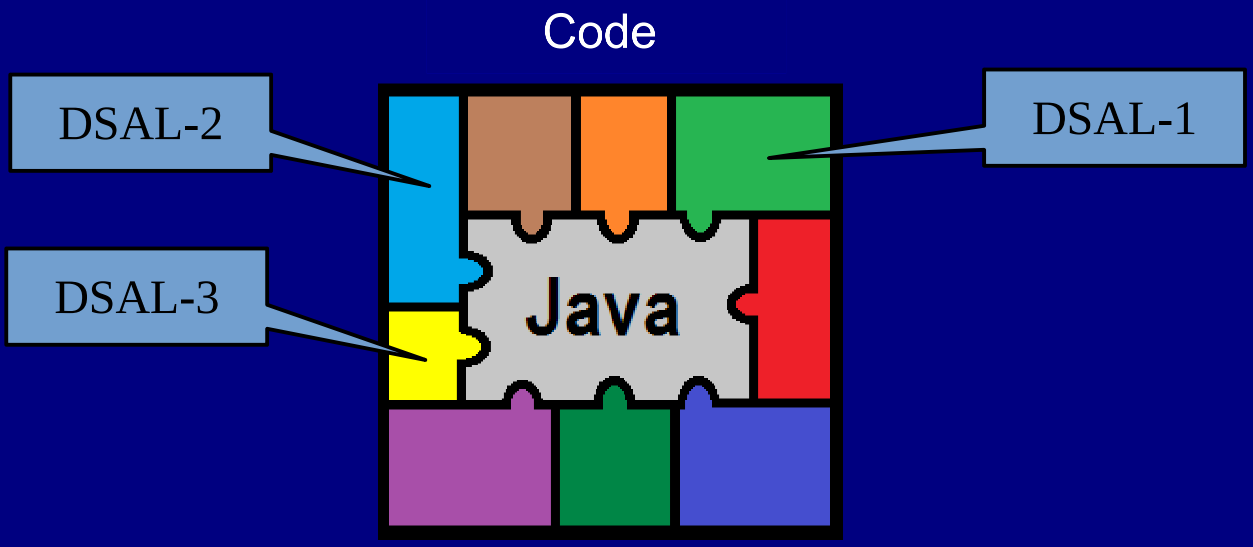


Code

**Java**

# Language Oriented Modularity (LOM)

- **A methodology that puts Domain Specific Aspect Languages (DSALs) at the center of the software modularization process.**
  - On-demand development and use of DSALs



# Pros of LOM

- Domain specific languages
  - Programming with more declarative and simpler languages than general purpose aspect languages (GPALs)
- Separation of crosscutting concerns
  - Improved software modularity compared to general purpose languages or DSLs

# Cons of LOM

- Cost
  - Definition and implementation cost is higher
- Effectiveness
  - Use of DSALs (compared to GPALs) is less effective than DSLs (compared to GPLs)

| | LOP & DSLs | LOM & DSALs |
|---|---|---|
| Cost-effectiveness | ☺ | ☹ |

# Application-Specific LOM

- Improve cost-effectiveness of LOM
  - Reduce the definition and implementation cost
  - More effective to use

| | DSALs | ASALs |
|---|---|---|
| Language Definition | 😐 | 😐 |
| Language Implementation | ☹ | ☺ |
| Language Use | ☹ | ☺ |

# Outline

- Introduction
- Application Specific LOM
- **Demonstration**
- Conclusion

# Synchronization in oVirt

- Prevents concurrent execution of conflicting commands
- Per-command configuration
  - Scattered across commands in oVirt-Engine
- Global locks handling
  - Tangled within CommandBase
- Problem in current design
  - Lack of traceability reduced productivity
  - And blamed directly for bugs

# Demo

- Developing a DSAL for synchronization in oVirt: https://youtu.be/PTy9rYDQSo4

# Outline

- Introduction
- Application Specific LOM
- Demonstration
- Conclusion

# Related Work

- **Language Workbenches**
  - **[Fowler, 2005] Language workbenches: The killer-app for domain specific languages.**
  - **[Lorenz and Rosenan, 2011] Cedalion: A language for language oriented programming.**
- **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.**

# Summary

- Crosscutting concerns (still) prevails software modularity in modern projects
  - Found several crosscutting concerns in oVirt
- Language oriented modularity
  - In theory, enjoy both worlds of DSLs and AOP
  - In practice, not practical
- Application specific LOM
  - Use ASALs instead of DSALs
- Evaluating application specific LOM in oVirt
  - Improve the modularity of oVirt using ASALs
  - More cost-effective LOM

# 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