



From the RCP Book To Reality

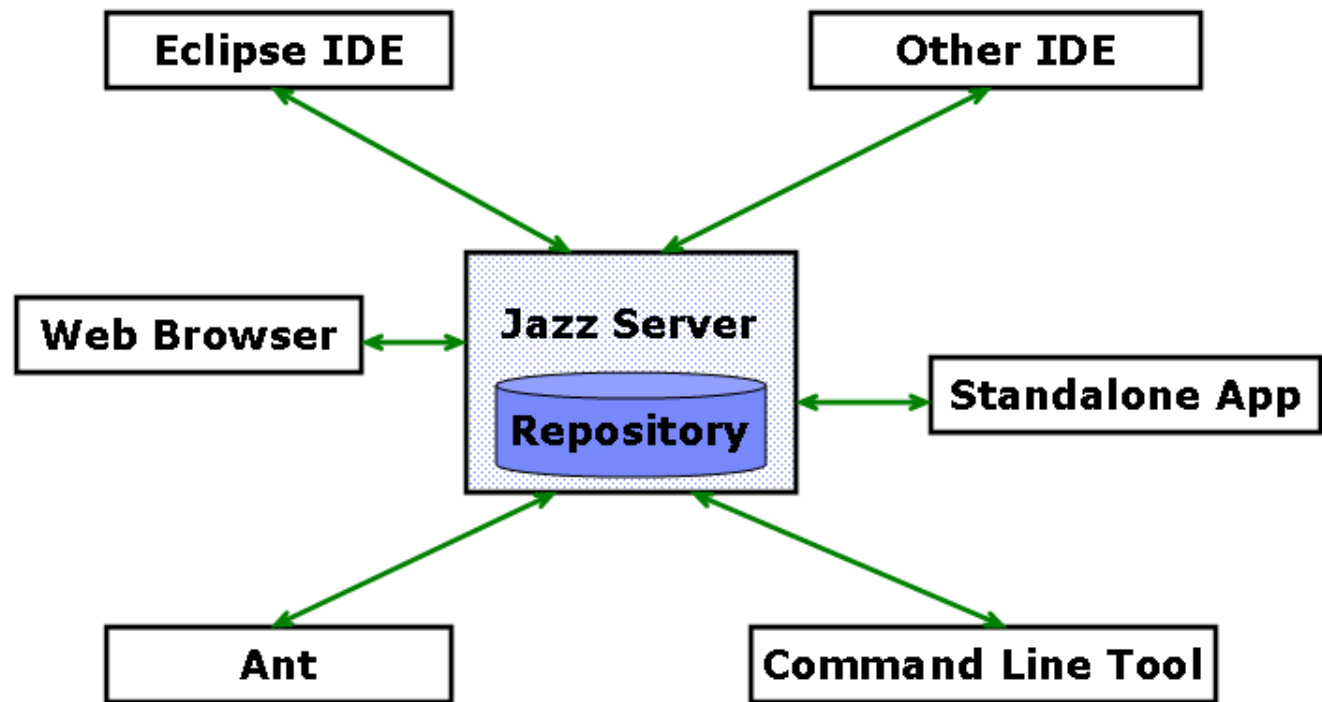
Jean-Michel Lemieux
IBM Rational, Ottawa, Canada

Past, Present, Future

- Eclipse committer since it's inception on core and team components.
- Wrote the book with Jeff.
- Working on a product to provide a collaborative team based platform and tooling.

Blank Slate – The basic strawman

- We knew that we needed a server and repository.
- We knew that we need different clients.



Blank Slate – What we imagined

- Team first
- Configurable through process
- Easily extensible
- Multiple clients
- Scalable
- Secure
- Standard middleware platform

Server Side

- We wanted an extensible server
 - ◆ Extension points and OSGi
- We wanted an extensible repository
 - ◆ EMF is a good way to describe a simple DDL
 - ◆ Good tooling available for codegen from EMF.ecore models
- We need a fast persistence storage
 - ◆ All custom code, nothing available from Eclipse in this space
 - ◆ OR mapping is too naive – solution is a combined OR+Document+Lucene
- We wanted productive development environment
 - ◆ Jetty, Derby, Debug and tooling

Server Side and Eclipse – Understanding jazz.war

Fits nicely with the Eclipse runtime.

/jazz/* →

/jazz/service/* →

../IWorkItemService

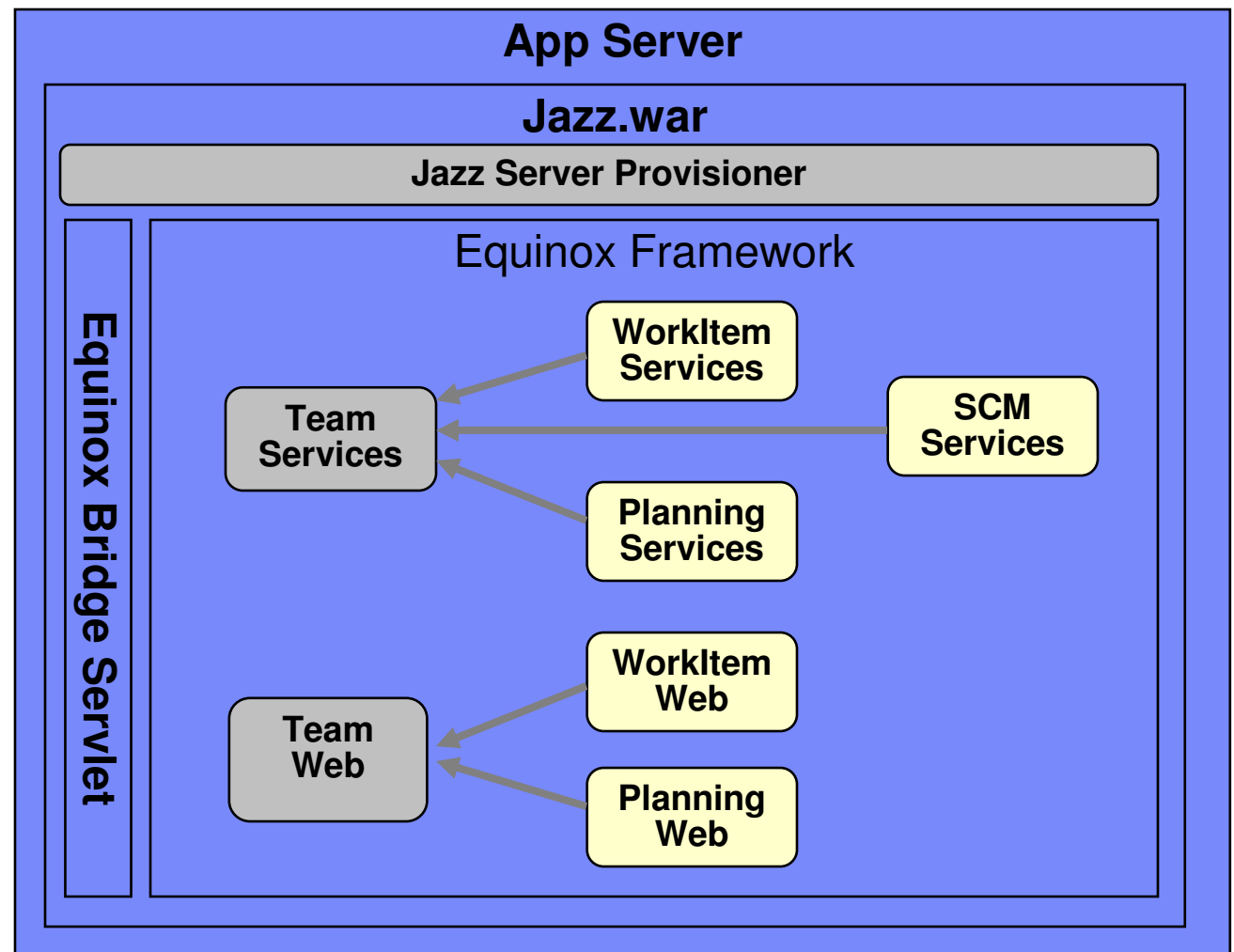
../IFileSystemService

../IPlanRestService

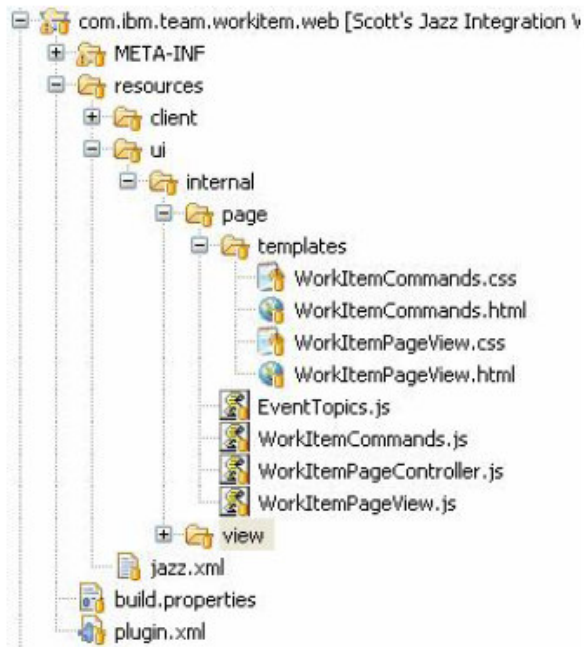
/jazz/web/* →

../WorkItemView.js

../PlanView.js



Extensibility Addicts – Plug-ins for Javascript



WorkItemPageView.js:

```
/*
 * The main view for the work item web UI. Responsible for displaying left-nav
 * content and for managing the main content area.
 */
dojo.widget.defineWidget(
  "com.ibm.team.workitem.web.ui.internal.page.WorkItemPageView",
  dojo.widget.HtmlWidget,
  {
    //
    // Other work item code may call these properties and methods.
    //

    /*
     * Returns the internal id of the widget currently displayed in the
     * page's content area.
     */
    getCurrentWidgetId: function() {
      return this._currentMainWidgetId;
    },
    /*
     * Returns the work item object currently displayed in the page's
     * content area.
     */
    getCurrentWorkItem: function() {
      return this._multipaneContentWidget.getCurrentWidget()._workItem;
    }
  }
);
```

Extensible Web UI

- Re-use Eclipse perspective concepts in the web.

```
<!-- work item web UI bundle registration -->
<extension point="com.ibm.team.repository.web.webBundles"/>

<!-- work item web UI page definition -->
<extension point="com.ibm.team.repository.web.basicPerspectives">
  <basicPerspective
    id="com.ibm.team.workitem.web.WorkItemPage"
    widget="com.ibm.team.workitem.web.ui..WorkItemPageView"
    name="Work Items"/>
</extension>
```

Extensible server

- Allow contributing services accessible over http to the server.

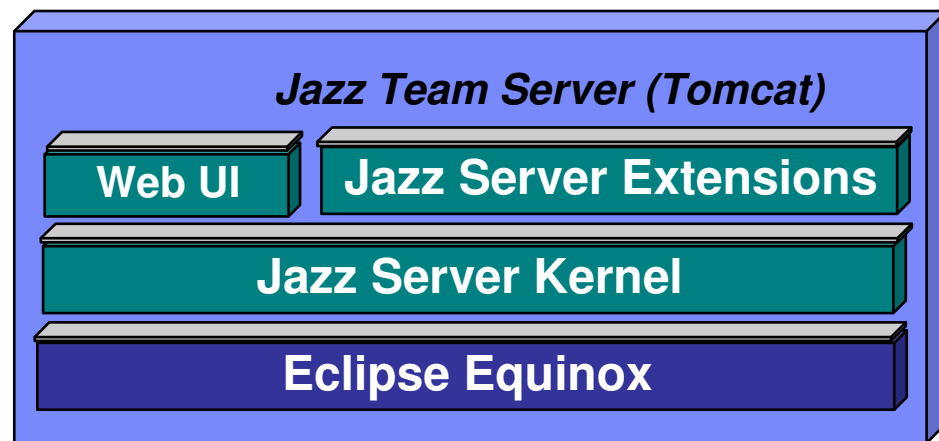
```
<extension point="com.ibm.team.repository.common.components">  
  <component  
    id= "com.ibm.team.foundation.wiki"  
    name= "Wiki"  
    packageURI= "com.ibm.team.foundation.wiki">  
    <service  
      name="Wiki RCP Service"  
      interfaceClass="com.ibm.team.wiki.IWikiService"/>  
  </component>  
</extension>
```

Server Summary

- In some ways simpler than the client because we have full control over runtime.
 - ◆ But still have to play well with many DBs.
- Missed the boat with service APIs
 - ◆ Custom (EMF serialized over http) but should of standardized the data formats for easier cross language interop.
- Eclipse based server provisioning is very interesting
 - ◆ But how about synchronized client/server provisioning
 - ◆ Client and server mismatches
- High value in running in standard web application contexts.

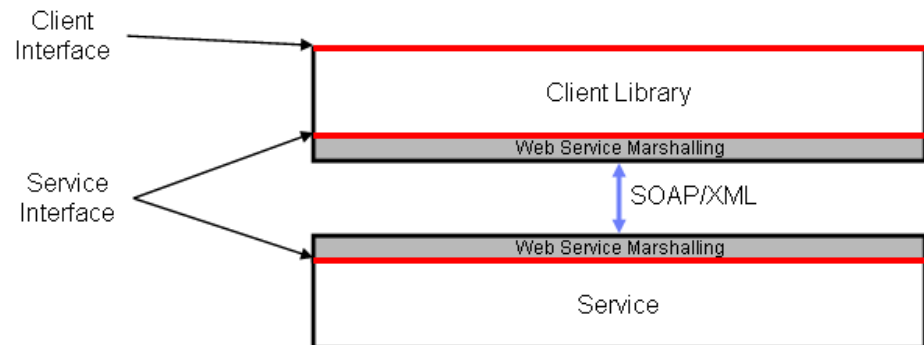
On the Client

- Support different languages, IDE integrations
- Plain java
- Extensibility and APIs



Java Client Architecture

- Common code
 - Code which runs on the server and on the client
 - Provides both the data models, shared business logic, and the service APIs
- Plain java
 - Wanted to make the server as accessible as possible
 - Callable from Ant via a standard Ant task.
 - Easy snippets without having to teach plug-ins
 - Still wanted extensions, adapter registry, emf package registry, and file locator.



Java Client Architecture

- What plug-ins in Eclipse are plain java compatible?
 - ◆ No references to OSGi
 - ◆ Eclipse has an important set of useful libraries that add to OSGi but are standalone
 - Registry
 - Jobs
 - EMF
 - ...
- Common and Client library plug-ins must be plain java compatible.
- Verified via plain java JUnits

Required Plug-ins

Specify the list of plug-ins required for t of this plug-in.

- org.eclipse.equinox.common
- org.eclipse.core.jobs
- org.eclipse.emf.ecore
- org.eclipse.emf.ecore.sdo
- com.ibm.team.repository.common
- com.ibm.team.repository.client
- com.ibm.team.foundation.common
- com.ibm.team.scm.common
- com.ibm.team.process.common
- com.ibm.team.process.client
- org.apache.commons.logging
- org.eclipse.core.runtime
- org.eclipse.osgi.util

Java Client Architecture

Extending the
Eclipse registry

```
public class StandaloneExtensionRegistry implements IRegistryProvider {  
    private final static Log logger = LogFactory.  
    private Object masterRegistryKey = new Object();  
    private Map<String, ResourceBundleClassLoader> pluginIdToBundlePropertiesLoader;  
    private IExtensionRegistry registry;  
    private class ResourceBundleClassLoader extends ClassLoader {  
    protected StandaloneExtensionRegistry() {  
        RegistryStrategy strategy = new RegistryStrategy(null /* don't cache the registry */,  
        registry = RegistryFactory.createRegistry(strategy, masterRegistryKey, null /*  
            * don't support user based  
            * contributions  
            */);  
        this.pluginIdToBundlePropertiesLoader = new HashMap<String, ResourceBundleClassLoader>(64);  
        Enumeration extensionFiles;  
        try {  
            extensionFiles = StandaloneExtensionRegistry.class  
                .getClassLoader().getResources("plugin.xml"); //$NON-NLS-1$  
        } catch (IOException e) {  
            //It will be very hard to carry on in this state  
            throw new RuntimeException(e);  
        }  
        int i = 0;  
        while (extensionFiles.hasMoreElements()) {  
            URL pluginXml = (URL) extensionFiles.nextElement();  
            processPlugin(registry, pluginXml, i);  
            i++;  
        }  
    }  
}
```

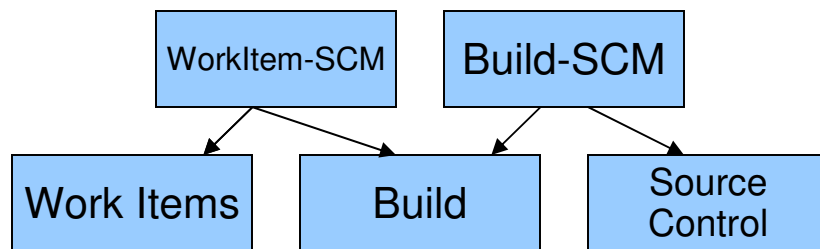
Java Client Architecture

- Don't assume client's are running or familiar with the Eclipse runtime (OSGi).

```
TeamPlatform.startup();  
ITeamRepositoryService srv = TeamPlatform.getTeamRepositoryService();  
ITeamRepository repo = srv.getTeamRepository(myRepositoryURL);  
repo.login(progressMonitor);  
IQueryClient qryClnt = repository.getClientLibrary(IQueryClient.class);  
IQueryResult result= qryClnt.findAll(myQueryExpression);  
processWorkItems(result);
```

Beyond Plain Java

- All other best practices adopted from Chapter 23, RCP everywhere.
- Split between RCP and Product plug-ins to enable building different products from the same code base.
- Split inter-component dependencies into bridge plug-ins to enable building sub-configurations of our product.



Client Plug-in Architecture is important

- How can we ensure that architecture isn't mistakenly broken.
- Our cool tools make it easy to break
 - ◆ Change a MANIFEST.MF file
 - ◆ Change a feature

UI Integrations

Protection of the UI and responsive threads

- ◆ ThreadCheck
 - Ensure that long running operations don't block the UI thread. Utility which protects response threads.
- ◆ Dynamic proxies and control over long running operations
 - Allows the client platform to describe and control over-the-wire services and tool appropriately.

Dynamic Proxies

- We control all calls to the server.
- Can mark thread as long running.

```
static Object newServiceInterfaceProxy(Class<?> interfaceClass,
    Object serviceImplementation, TeamRepository repository) {
    return Proxy.newProxyInstance(interfaceClass.getClassLoader(),
        new Class[] { interfaceClass },
        new ServiceInterfaceProxy(serviceImplementation, interfaceClass.getName(), repository));
}

protected ServiceInterfaceProxy(Object delegate, String serviceName, TeamRepository repository) {
    this.delegate = delegate;
    this.serviceName = serviceName;
    this.repository = repository;
}

public Object invoke(Object proxy, Method method, Object[] args)
    throws Throwable {
    Object result;
    long start = System.currentTimeMillis();
    boolean connectionError = false;
    try {
        ThreadCheck.checkLongOpsAllowed();
        TeamServiceCallContextImpl.checkCancelable();
        validateServiceCall(method);
        result = invokeServiceCall(method, args);
        // if the repository was logged-in but the last request failed, reset
        // the error state indicating that the connection is in a better
        // state.
        repository.setErrorState(ITeamRepository.ERROR_STATE_NONE, null);
    } catch (InvocationTargetException e) {
```

ThreadCheck

- Start by protecting the UI Thread
- Then call `longOpsProhibited()` on each responsive thread before trying to call the server.

```
/**
 * Boolean-valued thread-local variable indicating whether it is illegal to
 * call a long-running operation (ones tagged &at;LongOp) from a given
 * thread. (The ThreadLocal maintains a Boolean for each different thread.)
 * Initially Boolean.FALSE.
 * <p>
 * Note that the thread-local variable is declared private so that it can
 * only be accessed by this class's methods.
 * </p>
 *
 * @ShortOp This is a short operation; it may block only momentarily; safe
 *           to call from a responsive thread.
 */
private static ThreadLocal longOpsProhibited = new ThreadLocal() {
    @Override
    protected Object initialValue() {
        return Boolean.FALSE;
    }
};

/**
 * Returns whether it is illegal to call a long-running operation (ones
 * tagged &at;LongOp) from the calling thread. Initially <code>>false</code>.
 *
 * @return <code>>true</code> if long-running operations are prohibited,
 *         and <code>>false</code> if long-running operations are allowed
 * @ShortOp This is a short operation; it may block only momentarily; safe
 *           to call from a responsive thread.
 */
public static boolean longOpsProhibited() {
    return ((Boolean) (longOpsProhibited.get())).booleanValue();
}
```

Other languages, the future

Ruby

```
require 'jazz'

def printMatchingContributors(repo, filter, value)
  contributors = repo.contributors.contributors(filter, value)
  puts "#{contributors.length} contributors matching #{filter}="
  i = 0
  contributors.each do |contrib|
    puts "#{++i}: userId: #{contrib.userId}"
    puts "  name: #{contrib.name}"
    puts "  emailAddress: #{contrib.emailAddress}"
    puts "  archived: #{contrib.archived}"
  end
end

location = ARGV[2] || "https://localhost:9443/jazz/"
username = ARGV[0]
password = ARGV[1]

puts "connecting to #{location} as #{username}/#{password}"

repo = Jazz::Repository.new(location, username, password)
printMatchingContributors(repo, :userId, "me")
printMatchingContributors(repo, :name, "ADMIN")
```

C#

```
[TestMethod]
[Owner("Jean-Michel_Lemieux@ca.ibm.com")]
public void TestCreateStream() {
    // Setup
    TestProjectArea testProjectArea = getSharedTestP

    // create a stream with a name, description, and
    ParamsPostWorkspace postWorkspaceParams = new Param
    postWorkspaceParams.repositoryUrl = repo.Connecti
    postWorkspaceParams.name = "New Stream";
    postWorkspaceParams.description = "This stream is
    postWorkspaceParams.isStream = true;
    postWorkspaceParams.teamAreaId = testProjectArea.
    WorkspaceDTO stream1 = daemonClient.postWorkspac

    Assert.IsNotNull(stream1);
    Assert.IsNotNull(stream1.Workspace);

    ParamsGetWorkspace paramsGetWorkspace = new ParamsG
    WorkspaceDTO stream1Clone = scmService.getWorksp

    Assert.IsTrue(stream1Clone.Workspace.ItemId.Equa
```

Summary

- In reality the RCP book only covered a small portion of what I needed to know... **reality is complicated.**
- Be infected by the Eclipse “memes”
 - Design for extensibility.
 - Build a community around your products.

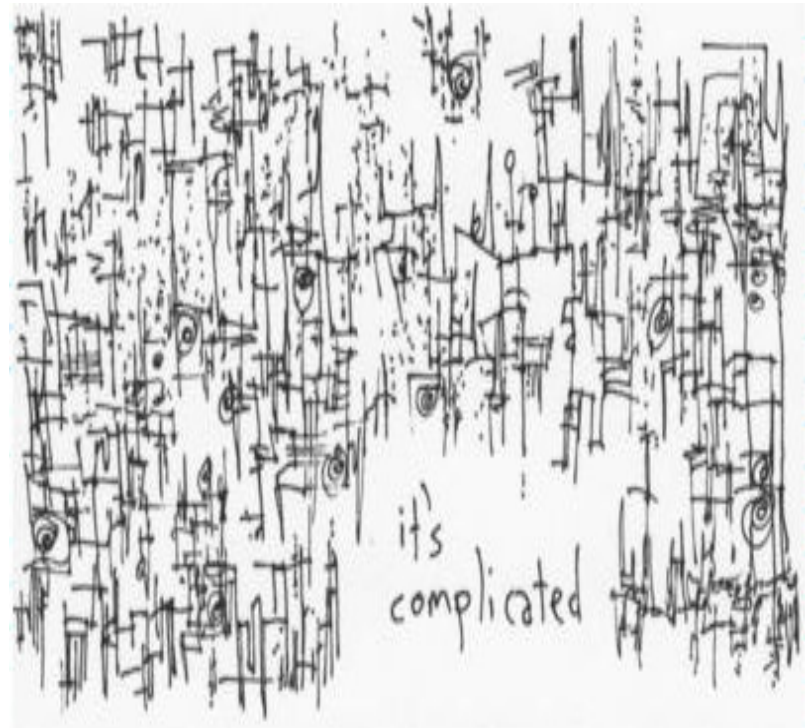


Image courtesy of [GapingVoid](#)