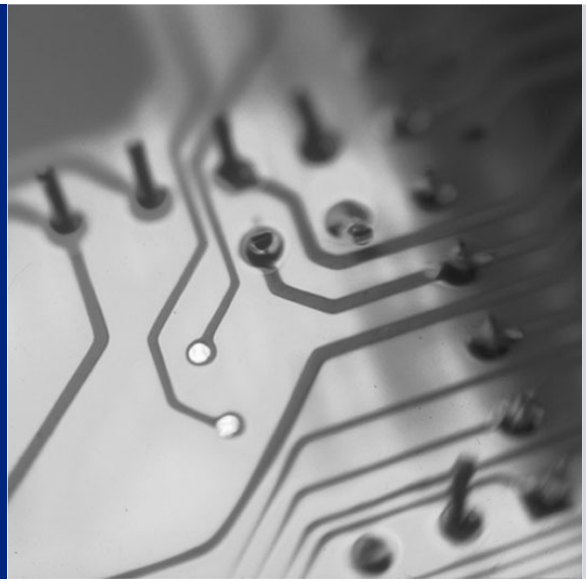


JCo Quick Summary

The relationship between Web Dynpro model objects, JCo Connection Pools and ABAP sessions within an SAP system

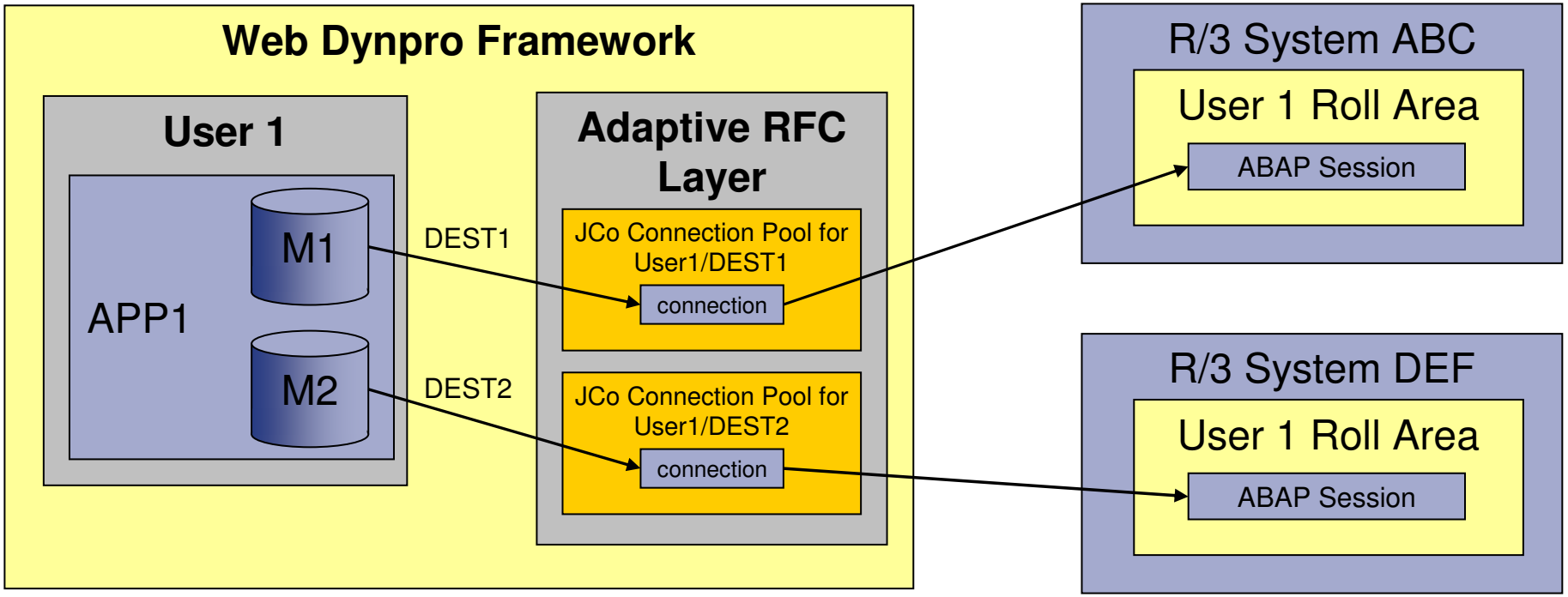
THE BEST-RUN BUSINESSES RUN SAP

Assumptions

The following assumptions have been made about the system configuration used in the following diagrams:

- **All user authentication is performed using single sign-on tickets (SSO2 Cookies)**
- **Web Dynpro application 1 (APP1) uses model objects M1 & M2**
- **Web Dynpro application 2 (APP2) uses model objects M3 & M4**
- **Model objects M1 and M3 use logical destination DEST1**
- **Model objects M2 and M4 use logical destination DEST2**
- **Logical destination 1 (DEST1) points to R/3 system ABC**
- **Logical destination 2 (DEST2) points to R/3 system DEF**

One user running one application

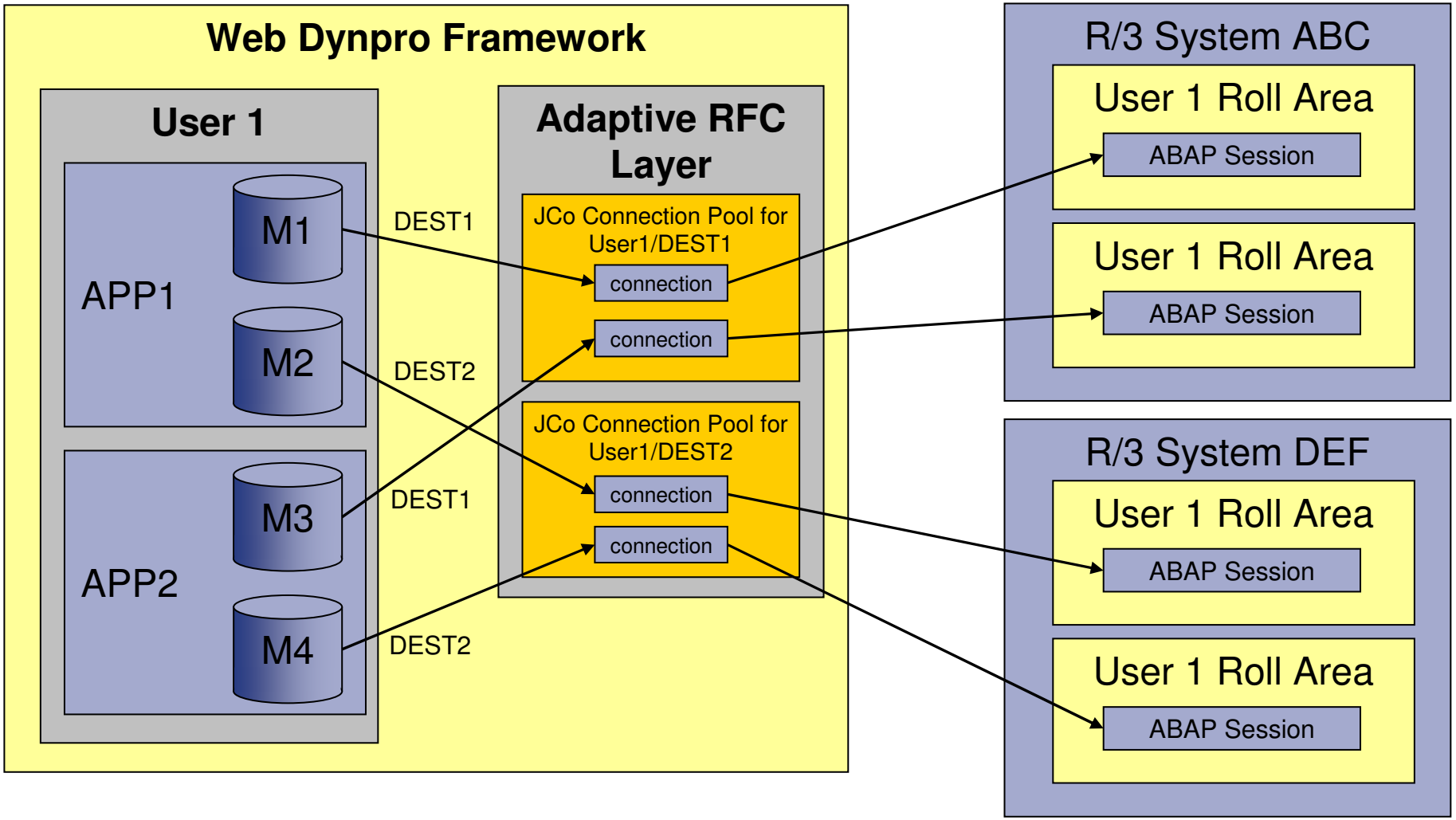


A correctly designed model will consume the minimum number of connections from the JCo connection pool.

Here, models M1 and M2 use different logical destinations because APP1 requires data from two different back end systems.

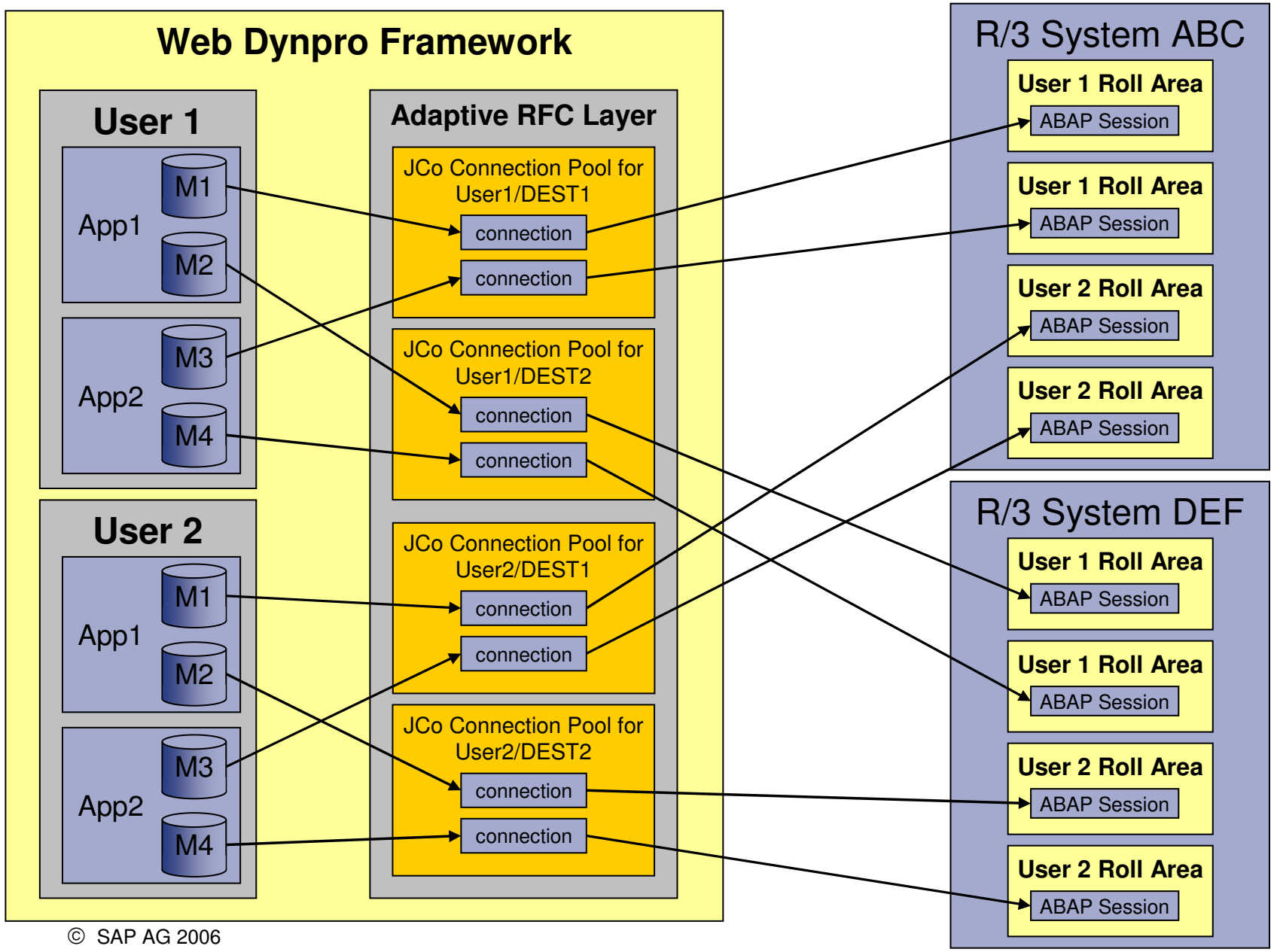
All the RFC function modules related to the business task(s) performed in system ABC are grouped together into model M1. Similarly, all the RFC function modules related to the business task(s) performed in system DEF are grouped together into model M2.

One user running two applications

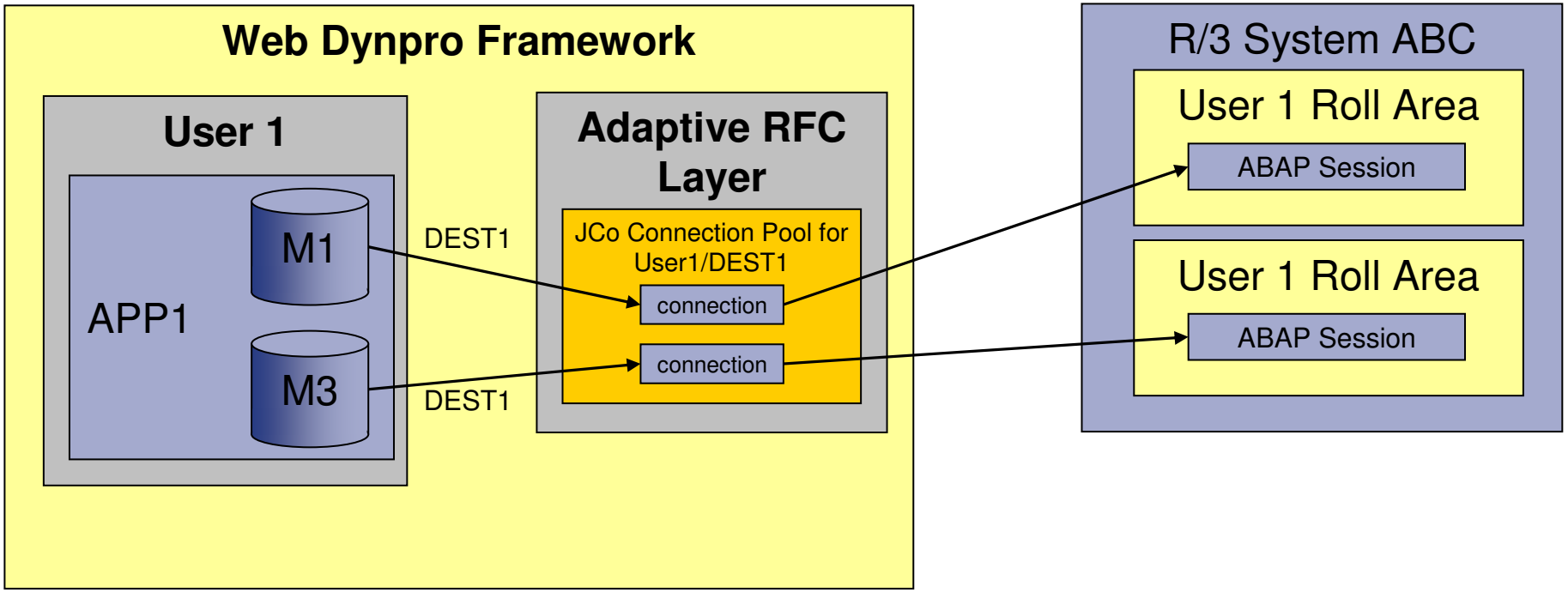


If the same user now starts APP2 (using models M3 and M4), this will result in connections being taken from the existing JCo pools, because the combination of user id and logical destination has not changed.

Two users running the same two applications



Inefficient model object design



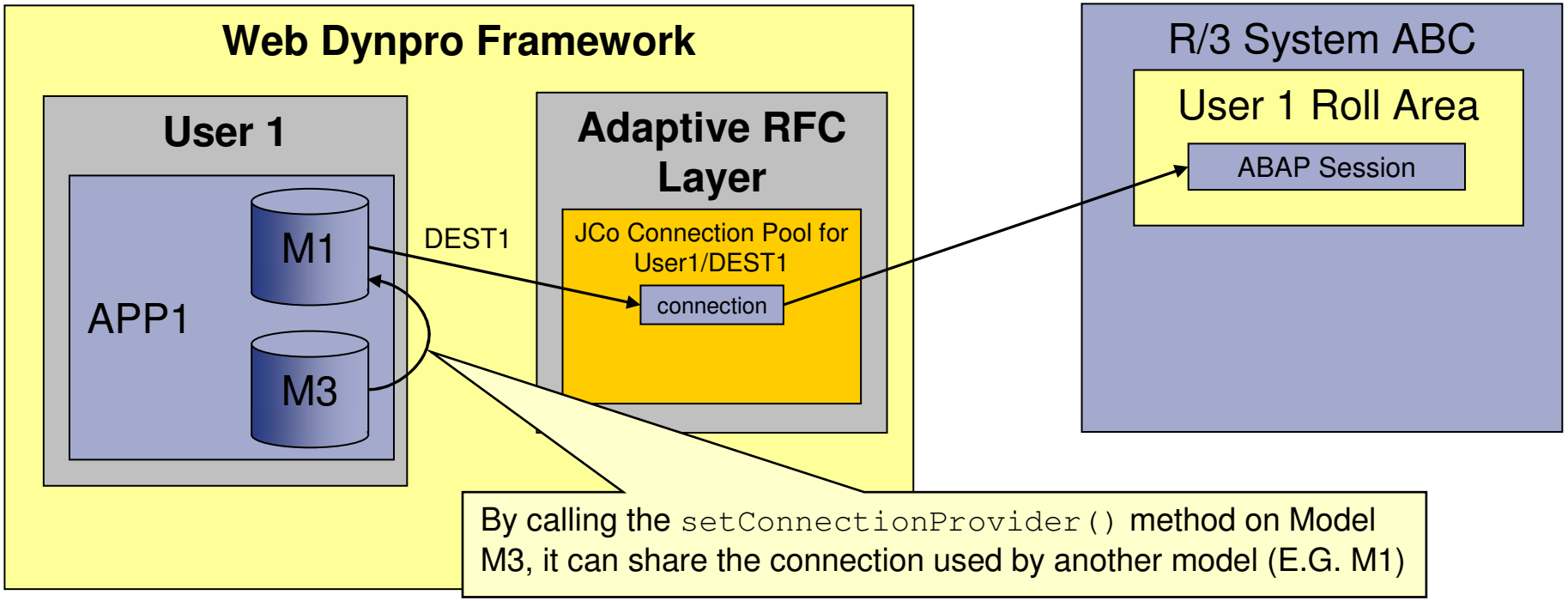
APP1 has now been modified to use models M1 and M3. Since models M1 and M3 both use the same logical destination, two connections will now be used from the JCo pool.

This will result in the same user logging on twice to the back-end SAP system.

Potentially, this could result in transactional processing problems in the SAP system if tasks performed by M1's function modules need to interact with the tasks performed by M3's function modules.

If this is the case, then either merge the model objects in M1 and M3 into a single model object, or use explicit connection management.

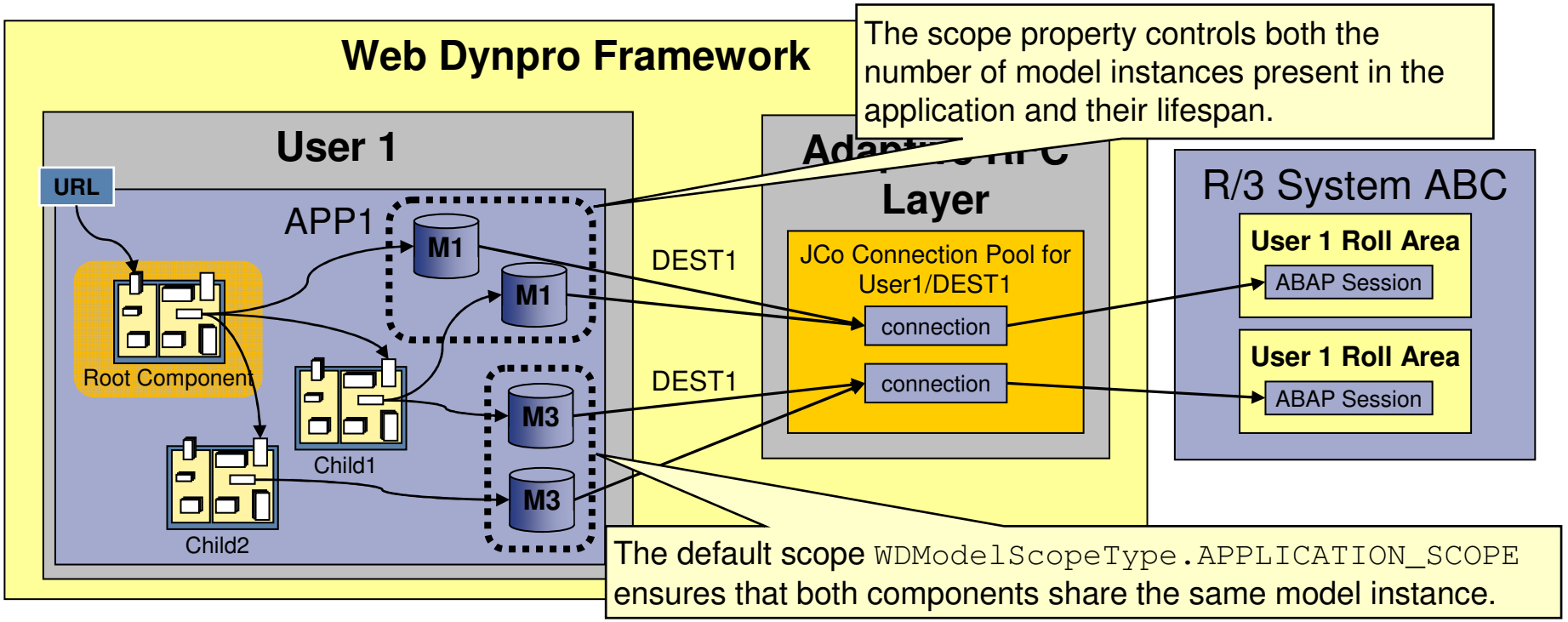
Explicit connection management



SAP recommends that, if possible, model objects should be merged since this simplifies connection management. However, SAP also recognises that it is not always possible or feasible always to merge model objects. Therefore explicit connection management is possible. This allows the connection used by one model to be shared with another model.

In this situation, the models M1 and M3 now share the same connection from the JCo pool. This means that the user only logs on once to the back-end system, and it also permits interaction between M1 and M3's function modules at a transactional level within the ABAP system.

Model instance scope



It is possible for the same model to be used by multiple components within the same application.

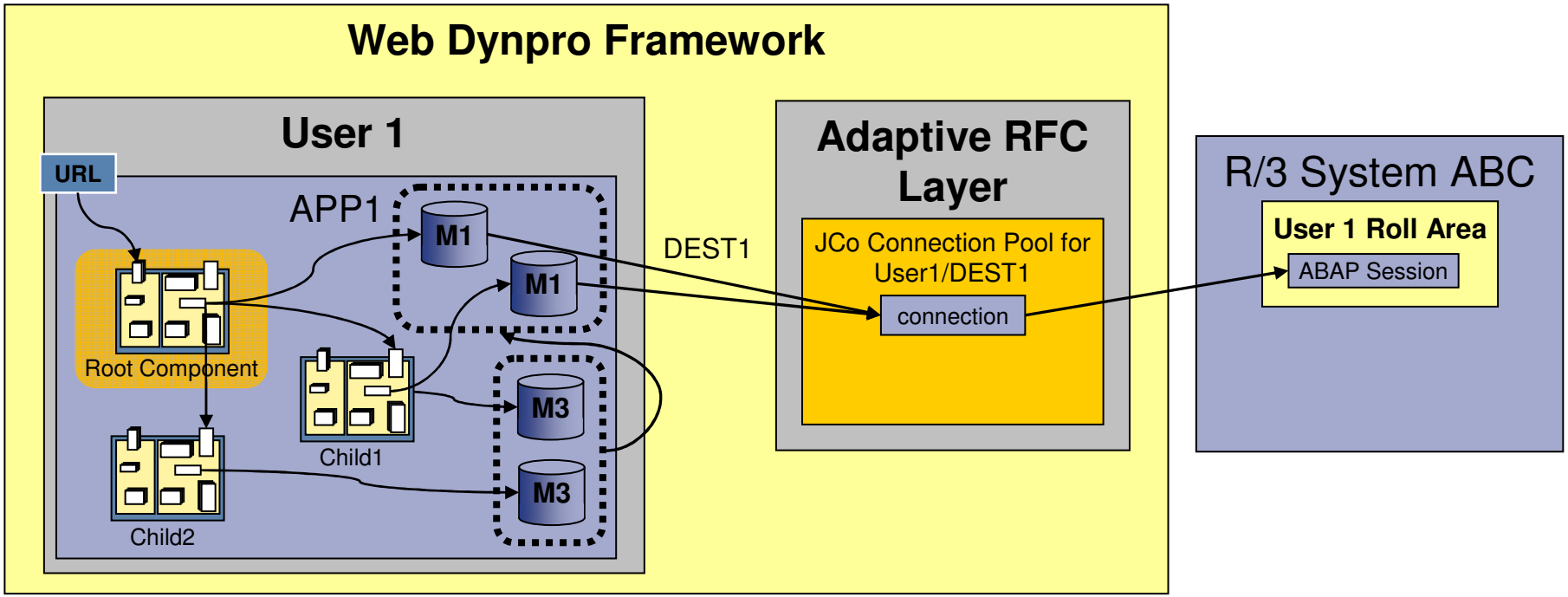
In this case, when the different components create their model instances, the scope property (which by default is set to `WDModelScopeType.APPLICATION_SCOPE`) will ensure that there is only model class instance even though two components will both be using that model. This in turn, ensures that JCo connections are shared.

However, even though the number of model classes has been optimized, each model class with still consume its own connection from the pool. This will result in the same user logging on to the back-end system twice.

Is this really necessary?

Well, it depends entirely on your application design, but the answer is usually “No”.

Optimized connection usage

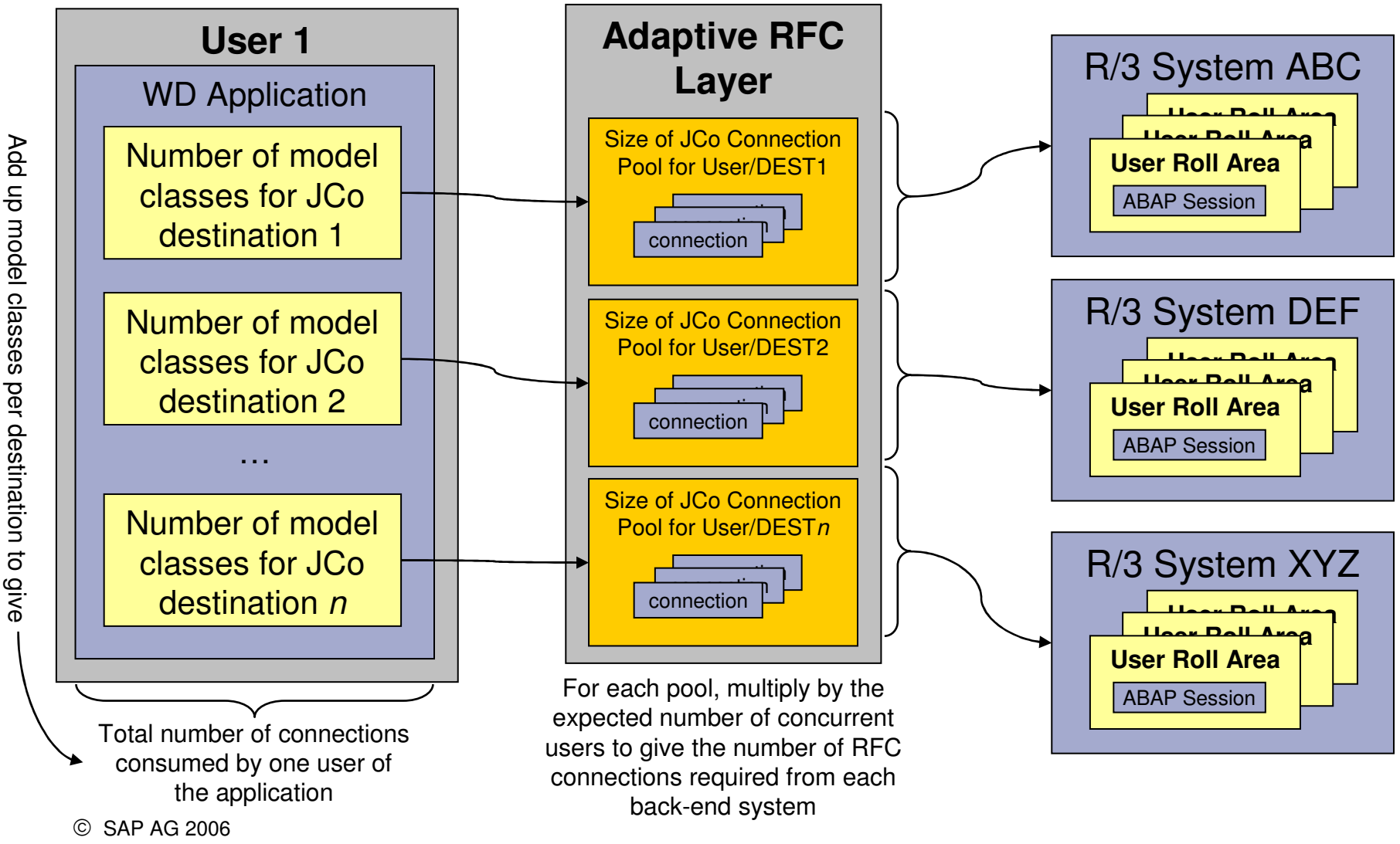


By additionally calling the `setConnectionProvider()` method for model class M3, you can ensure that the model class instance for M3 shares the same JCo connection as model class M1.

This will result in a single logon to the back-end SAP system.

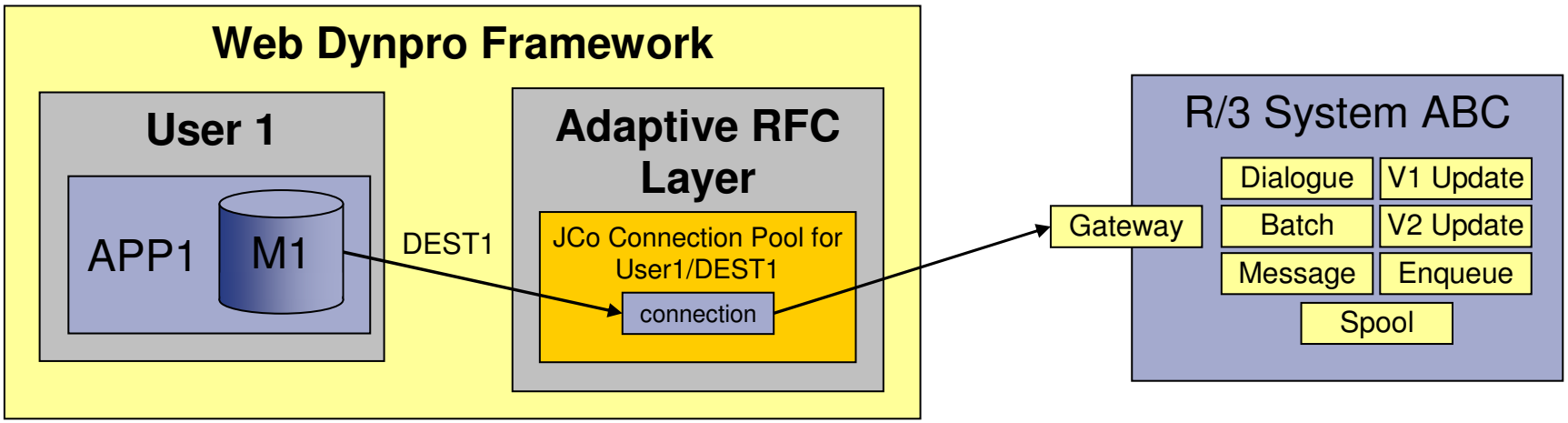
Calculating JCo Pool Sizes

This calculation assumes that ticket based authentication will be used. In other words, all pools required by one instance of the WD application are created for the same user.
This is only a basic approach to connection calculation and may not work for every application.



Configuring an SAP Gateway process – 1

All external programs wishing to invoke ABAP function modules do so by communicating with the SAP system through the Gateway process.

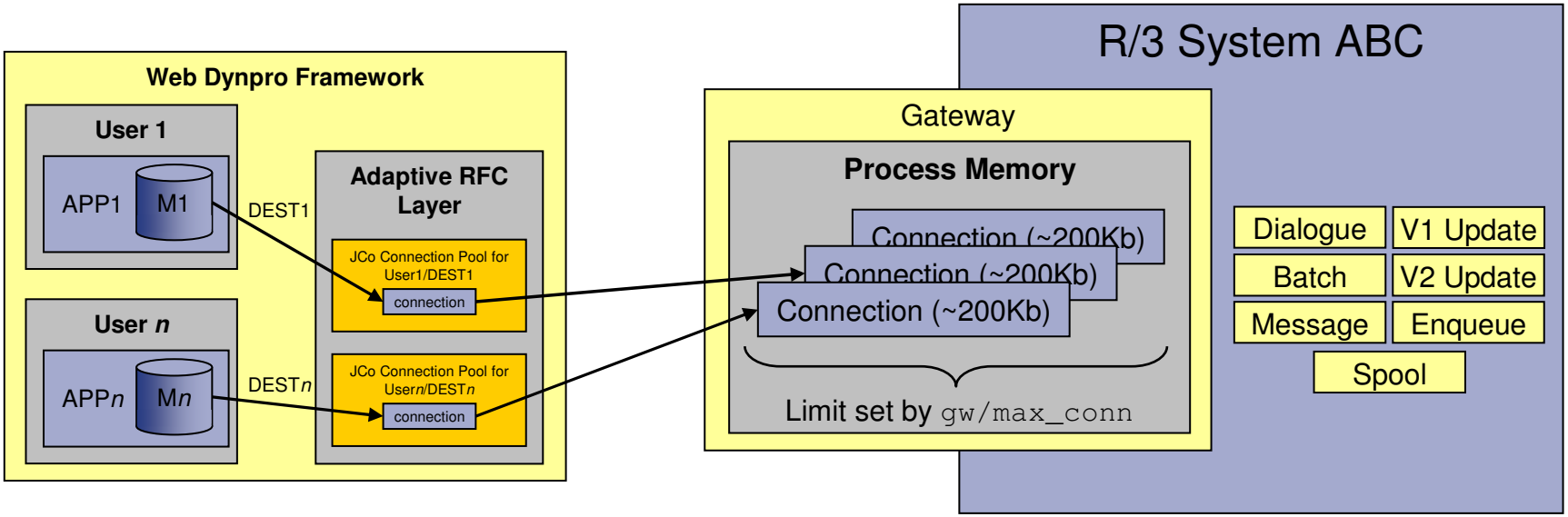


The SAP Gateway process must be configured correctly for the expected number of RFC connections.

If the gateway is not configured correctly, then under high work load situations, it is possible that the number of available connections could become exhausted. This will result in errors in the Adaptive RFC Layer (unable to obtain connection) which in turn, will appear as application errors in the Web Dynpro Application.

Configuring an SAP Gateway process – 2

Each active connection to the gateway requires approximately 200Kb of memory.



The SAP Kernel parameter `gw/max_conn` must be defined correctly in order to handle the maximum number of expected gateway connections (default = 500).

WARNING!

In any SAP system based on the 6.40 kernel or lower, the gateway has a hard limit of 2000 concurrent connections.

This limit is not present in 7.0 versions of the gateway.

Summary – 1

- Before a JCo destination can be created, you must first have created a Technical System in your System Landscape Directory (SLD)
- JCo destination names are case sensitive!
- JCo connection pools are created lazily. This means that the pool will not exist until you call the `execute()` method of the first model object using that pool.
- JCo connection pools are destroyed either when the user:
  - Formally logs off
  - Closes their browser
  - Issues a URL to a different website
  - Allows the application to time out.

Summary – 2

- JCo connection pools are created based on the combination of user id and JCo destination name.
- Connections within a pool are allocated on the basis of which model class is making the request.
  If models M1 and M3 are used by the same user for the same destination, then a single JCo pool will be created and from it, two connections will be consumed.
- In addition to the number of JCo connections required by the application connections, a metadata connection will also be required.
- The number of metadata connections required will be optimized according to the destination and user id.
  This optimization is performed across the entire Adaptive RFC layer; therefore, one connection will be shared by *all* models, in *all* applications, connecting to the *same* backend system with the *same* user id.

Summary – 3

- There is no single calculation for pool size requirements that will work in all situations; however, the following formula will be a good starting point (assuming that explicit connection management is *not* being used).
  - For each JCo destination, add up the number of model classes that will be created. This gives the number of connections that will be required from a single pool.
  - By adding up the connections required by all the pools, you arrive at the total number of JCo connections consumed when one user runs the application.
  - For each JCo destination, multiply the number of connections in the pool by the number of expected concurrent users, and this gives the minimum number of RFC connections that your back-end SAP system must provide.
  - Check with your Basis Administrator that the Dispatcher RFC parameters (`rdisp/rfc_*`) and Gateway parameters (`gw/*`) are configured correctly for the expected work load.

Summary – 4

- At runtime, the gateway process in the SAP system manages all active RFC connections.
  - The gateway process can be monitored using transaction `SMGW`
- The memory consumption of the gateway process is proportional to the number of active RFC connections
- For high workload environments, the `gw/max_conn` parameter should be adjusted (default value = 500):
  - **Important!**
    Setting `gw/max_conn > 2000` will have ***no*** effect for a gateway process of kernel version <= 6.40!