Note: As part of a service revision, Fujitsu will not newly accept the Customer's Service Application for the PF Service after 00:00 (UTC) on 11th January, 2018 (Thursday). The timeframe for restarting the acceptance of new service applications will be notified separately. *The contents of this document are subject to change after the major service revision.
Contents

- About PF
- Development of applications that meet the demands of mission-critical business
  - Standardization of application architecture
  - Modeling individual business functions as services
  - Design quality assurance
  - Business rule definition
  - Implementation of thorough impact analysis
  - Simple implementation of recovery, fault analysis, and external linkage
    - Data assurance (recovery) function
    - Data store function
    - Performance bottleneck analysis function
    - External service linkage function
  - Supplied plug-ins

- Rapid, robust construction and deployment of an execution platform
  - Application execution platform
  - User resource management
  - Monitoring of operational status; operation
  - Log management
  - Database
  - Patch application
  - Access Control
  - Manual scaling
  - Log monitor setting/ email recipient setting
  - Web API

- Explanation of billing model
- Restrictions and Notes

Attachment: System Configuration Package List
What is PF?

The PF Service allows legacy assets to be modernized, realizing a robust system responsive to changing business requirements.

- **Supports the development of applications that meet the demands of mission-critical business**
  Drastically reduces development timelines, improves maintainability and operability, and shrinks development scale.

- **Supports rapid, robust construction and the deployment of execution platforms**
  Systems can be easily built by simply selecting the execution environment definition and deploying the resources you have developed.

### Development of applications that meet the demands of mission-critical systems

- Use case
- Method for linking design elements
- Database design
- Traceable from design → implementation
- Takes operation into account
- High quality CI/CD with short cycle times

### Rapid, robust building and deployment of an execution platform

- Selection of execution environment definition
- Deployment of developed resources
- Construction complete ⇒ User access
Development of Applications that Meet the Demands of Mission-critical Business

The development methods and rules provided by PF enable development of applications with a high level of maintainability and operability.

- **Features of application development using PF**

  - **Standardization of Application Architecture**
    - Standardizing the type of application architecture and managing the performance architecture at design time realizes an application architecture without any gap between design and implementation.

  - **Modeling Individual Business Functions as Services**
    - Manages design and implementation using the smallest unit of a business function, which is a service. Using models and rules to implement service functions, and defining and managing the linkage flow between services improves service portability.

  - **Design Quality Assurance**
    - Maintaining the relationship between changing design information ensures design quality.

  - **Business Rule Definition**
    - Using design tools to define the business rules that establish the principles that inform business decisions, and using a rules engine to evaluate these rules simplifies the normalization of conditions and results.

  - **Implementation of Thorough Impact Analysis**
    - Using design tools and a code checker for Java resources to strictly regulate standardization constraints enables thorough impact analysis at resource modification time.

  - **Simple Implementation of Recovery, Fault Analysis, and External Linkage**
    - Facilitates implementation of data assurance, performance bottleneck analysis, data store linkage, and external service linkage.
A highly maintainable system is a change-tolerant system, and for this you need **scalability** and **flexibility**.

- **Scalability** (= Built in at design time)
- Predict the functions you want to add or change, and achieve this at minimal cost and in minimal time.
- **Flexibility** (= Aided by technology)
- Realize unforeseen function additions or changes at minimal cost and in minimal time.

**Maintainability demands:**

- **High productivity**
- Represent your business architecture as a model
- **Rules** (Principles)
- Draw on principles and imperatives (≠ phenomena), distilling the essential qualities into rules
- **Model** (Concept design)
- Architecture that doesn’t require manual developments

**Prevent structural collapse**

- **Rust resistance**
- Prevent system rust-out
- **Structural collapse**
- Identify the places in which functions will be implemented
- **Asset bloat**
- Prevent asset bloat
- **Tangled relationships**
- Avoid having duplicate logic scattered throughout
- **Stay tangle-free and traceable**

Reference: Conceptual Structure for Ensuring Maintainability
Standardization of Application Architecture (1/2)

- Realizing high maintainability and high productivity
  Having a standardized application architecture and robust business application architecture enables development of highly maintainable, high-quality applications. The design content can be generated without modification as definitions and execution resources, enabling application development that reduces the development load and boosts productivity.

- Application architecture can be standardized by defining models, use cases, and business rules.

Output is determined by how well the model, use case, and business rules are organized

Example: Extracted design pattern

<table>
<thead>
<tr>
<th>Business list</th>
<th>Telecommunications carrier use case</th>
<th>Main rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard contract (service start)</td>
<td>○</td>
<td>Perform new subscription</td>
</tr>
<tr>
<td>Amendment of licensee information</td>
<td>○</td>
<td>Change licensee</td>
</tr>
<tr>
<td></td>
<td>○</td>
<td>Change licensee information</td>
</tr>
</tbody>
</table>

Change of name, change of address, etc.

Extract elements

1. Rule description
   - Rule ID: xxx 10001
   - Rule name: Name can/cannot be changed
   - Rule type: Name
   - Description: Determines whether customer name can be changed

2. Input & output information
   - Input information: Licensed ID, licensee surname after change, licensee name after change
   - Output information: Whether name can be changed: 1) Can be changed 0) Cannot be changed

Customer contract model

Business list

(Business) Model

- Group
- Licensee
- User
- Biller

- Basic contract
- Price plan
- Discount service
- Telephone no.
- Cell phone

Use case (API)

- Behavior
  - User
  - Other systems
  - Quasi-normal, or abnormal

(Business) Rules

Conditions

Behavior
Using design tools to automatically generate definitions and execution resources

1. An application is **highly maintainable** when its architecture is standardized and there is no gap between design and construction, making it robust.

2. Using design content as-is as an execution resource reduces development load and **enhances** application development **productivity**.
Improving service portability

You can manage design and implementation using the smallest unit of a business function, which is a service. Using models and rules to implement service functions, and defining and managing the linkage flow between services improves service portability. Using business process flow control to create associations between multiple services also enables provision of services with a higher level of granularity.

Example: Using PF to model the service start task as a service

Contract service

Service creation and start service

Model data

Rules engine

Service acceptance content
### Design Quality Assurance

- **Ensuring design quality**

  Ensure design quality by maintaining the relationship between changing design information.

- **Provides a design plug-in to ensure the relationship with design resources.**
  1. Centralized management of design information in the design plug-in enables regular updates of the relationship between changing design information.
  2. The relationship with affected design documents can be visualized hierarchically.

**Upstream design using Eclipse plug-in**

- **Requirements analysis**
  - Model design
- **High-level design**
  - Business flow design
  - Rule design
  - Web flow design
- **Detailed design**
  - Table design

**Maintain the relationships between information**

- **Design information**
  - XML

**Design resources analysis**

- **Business process flow**
  - Rules
- **Table**
  - Model

Ensure design quality by maintaining the relationship between changing design information.
Facilitating normalization of conditions and results

Using design tools to define business rules simplifies the normalization of conditions and results.

- The rules engine evaluates the data model content based on the rule definition, and returns a result.
  1. You can describe business data item names as they are in the rules definition, making rule content clearer.
  2. You can invoke a rules definition hierarchically from another rules definition, enabling standardization of like rules.

Business Rule Definition (1/2)
Rule content supports an evaluation criteria table and a decision table. It is easy to define rules content in accordance with business principles and imperatives.

**Tips for creating rules content**

1. Rules definition methods usually consist of an evaluation criteria table and a decision table, depending on rules content.
2. Place items with highly variable elements in a master.
3. Draw on principles and imperatives to simplify the normalization of conditions and results as much as possible.

**Example: Parking lot service specs**

<table>
<thead>
<tr>
<th>Table of fees applied</th>
<th>Fee Table A</th>
<th>Fee Table B</th>
<th>Fee Table C</th>
</tr>
</thead>
<tbody>
<tr>
<td>One parking period</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basic fee (per day)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tax inclusive</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1,200 (\text{yen})</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Up to (1_{\text{day}})</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From (2_{\text{days}}) to (3_{\text{days}})</td>
<td>1,000 (\text{yen})</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\geq 4) <em>{\text{days}}</em></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>800 (\text{yen}) (No upper limit)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Optional conditions**

1. Use of limited express train to/from station center ⇒ Mandatory condition
2. For a product spend of 30,000 \(\text{yen}\) or more ⇒ Shopping discount
3. When using a cinema within the center ⇒ Cinema discount
4. When using both (2) and (3) ⇒ Combo discount

**Rules content creation**

1. Ranking criteria applied to parking fee

<table>
<thead>
<tr>
<th>Result</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank A</td>
<td>Parking period 0-1 day</td>
</tr>
<tr>
<td>Rank B</td>
<td>Parking period 2-3 days</td>
</tr>
<tr>
<td>Rank C</td>
<td>Parking period &gt;3 days</td>
</tr>
</tbody>
</table>

2. Parking fee optional conditions (criteria)

<table>
<thead>
<tr>
<th>Condition</th>
<th>Pattern specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use of limited express train to/from station = “Yes”</td>
<td>N Y Y Y</td>
</tr>
<tr>
<td>Product spend &gt;= 30,000 (\text{yen}) = “Yes”</td>
<td>- Y N Y</td>
</tr>
<tr>
<td>Cinema use = “Yes”</td>
<td>- N Y Y</td>
</tr>
<tr>
<td>Shopping discount</td>
<td>X</td>
</tr>
<tr>
<td>Cinema discount</td>
<td>X</td>
</tr>
<tr>
<td>Combo discount</td>
<td>X</td>
</tr>
</tbody>
</table>
Thorough impact analysis

Using design tools and a code checker for Java resources to strictly regulate standardization constraints enables a thorough impact analysis during resource modification.

Problems with existing impact analysis

1. The Eclipse and Java hierarchy viewers both have limitations imposed by messaging and data retention (conversion to XML, etc.)
2. Given the support for highly flexible naming of internal variables, etc., 100 points cannot be collected under the standard naming conventions. (*Input control imposed by JDT(*))
3. grep and other string searches require decisions on ineligible resources, leading to operational inefficiency.

Note: JDT (Java Development Tools): Tools feature for Java source compilation, code completion, etc.

Solve problems

A thorough impact analysis can be implemented using design tools, resource constraints imposed by a code checker, and impact analysis viewer.

Impact Analysis Viewer

![Impact Analysis Diagram]

- **Change element**: Represents a change or modification to the resource.
- **Related resource**: Indicates a relationship or dependency between resources.
- **Relationship**: Shows the connection between change and related resources.
- **Area identified by a user as requiring resource modification**: Highlights the specific areas that need attention for resource modification.
Impact analysis viewer

Thorough impact analysis is realized by using design tools to design the architecture to associate everything based on *model items*, making it self-explanatory as to *which function* is used *from where*, *by whom* and *for what purpose*. This enables the discovery of meaningful relationships, rather than merely the discovery of words, facilitating multi-level analysis.

A framework that associates everything based on model data ensures a correct understanding of the influence of each element.
Practical example of thorough impact analysis

1. Manage model item attributes as class variables by generating the class from the design information.
2. Business java: Use the Key class to manipulate model items.
3. The relationship between the checking of a model item and a variable is managed using a structure that sets an instance in an internal variable to ensure that the relationship is not severed.

**Eclipse plug-in**

- **Contract ID**
- **Contract type**
- **Licensee name**

**Item attributes**

- **Item name**: Contract type
- **Type**: Long
- **Min**: 0
- **Max**: 9999
- **Mandatory**: false
- **Space-fill to end**: false
- **Encrypt**: false

**Catalog class (generated by plug-in)**

```java
public class Keys {
    public interface Constructor {
        public static final String contractID = "contractID";
        public static final String contractType = "contractType";
    }
    public interface Keys {
        public static final String contractID = "contractID";
        public static final String contractType = "contractType";
        public static final String contractName = "contractName";
        public static final String contractModel = "contractModel";
    }
}
```

**Java class (MBC coding)**

```java
30 // Fetch the model to be manipulated (contract model)
31 Contract contract = (contract) this.getModelBase(Keys.contract);
32 // Set interface data value in model
33 contract.setValue(Keys.contractID, contractInput.getLong(Keys.contractID));
34 // Contract type in interface data
35 String contractType_input = contractInput.getString(Keys.contractType);
36 String str = contractInput.getString(Keys.contractType);
```

[E140-08]: Use either * model item name (IF data item name) or * model item name (IF item name)_* as the instance variable name. Compile fails due to error
Simple Implementation of Recovery, Fault Analysis, and External Linkage

- Facilitates implementation of data assurance, performance bottleneck analysis, data store linkage, and external service linkage. The following functions are provided:
  ① Data assurance (recovery) function
  ② Data store function
  ③ Performance bottleneck analysis support
  ④ External service linkage function
Data Assurance (Recovery) Function (1/3)

- Data assurance (recovery) function
  - Ensures consistency in transactions that include database update and message update at recovery time.
  - You can select Cancel recovery and Forward recovery.

If an error occurs during synchronous processing, you can skip subsequent processing and roll back the database to the data update.

Image: Cancel recovery control
If an error occurs during asynchronous processing, the asynchronous part can be re-executed simply by re-entering the event.

Forward recovery operation (continued on next slide)

1. An error occurs during asynchronous processing.
2. The system administrator is notified.
Forward recovery operation (continued from previous slide)

③ The system administrator uses a patch, etc., to correct the recovery log data.
④ The system administrator re-enters the event in which the error occurred.
⑤ The data assurance function is used to skip the part that completed normally and re-execute the synchronous part based on the recovery and execution logs.

The example shows data conversion using external service linkage, but a system internal asynchronous update is also possible.
Data Store Function

- Data stores can easily be switched without changing applications.
- Data can be encrypted and stored by specifying in the model definition that the model item is to be encrypted.

Data can be encrypted

Data stores can easily be switched
Query comment insertion during SQL query execution
Embedding a message ID in the SQL query makes it easier to identify choke points, thus reducing the hours of labor involved in investigating performance issues.

By comparing the EAW log against the SQL code that caused the performance bottleneck, you can trace the operation that occurred at SQL execution time, including the application that was executed, the time zone, and the server name.
Automatic execution plan output function
Outputting an execution plan during SQL execution makes it easier to identify problematic applications and SQL code, thus reducing the hours of labor involved in investigating performance issues.

The automatic execution plan output function allows you to immediately identify application performance bottlenecks and problematic SQL code.
External Service Linkage Function

- Provides a simple mechanism for data linkage with external services
- Provides a data conversion function and a request send/receive function
# Supplied Plug-ins

## List of the plug-ins provided

<table>
<thead>
<tr>
<th>Plug-in name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model and Rule Definitions</td>
<td>Defines the models, rules, the database tables connected to models, and the manipulations (search criteria, etc.) of those tables</td>
</tr>
<tr>
<td>SimpleEventFlow (BPM) Definition</td>
<td>Defines the execution sequence (flow control) of models, on a per-use-case basis</td>
</tr>
<tr>
<td>URI Mapping Definitions</td>
<td>Defines the connection between the external access REST interface and the application developed by the licensee</td>
</tr>
<tr>
<td>Code Checker</td>
<td>Checks whether Java resources conform to Java coding standards</td>
</tr>
<tr>
<td>Impact Analysis</td>
<td>Conducts analysis of model definitions, rules definitions, Java resources, etc., analyzing the potential consequences of resource modification</td>
</tr>
</tbody>
</table>
Rapid, Robust Construction and Deployment of an Execution Platform

Provides an application execution platform that can also activate mission-critical systems. Simply select the execution environment definition (system configuration package), deploy the resources you have developed, and the system build is complete. Functions are also provided to monitor the operational status and for browsing operation screens and logs.
Rapid, Robust Construction and Deployment of an Execution Platform

List of functions provided

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Application execution platform</td>
<td>Provision of system configuration packages Building of an execution platform</td>
</tr>
<tr>
<td>2</td>
<td>User resource management</td>
<td>Registration and deployment of developed resources</td>
</tr>
<tr>
<td>3</td>
<td>Monitoring of operational status; operation</td>
<td>Displays the system’s operational status (Start/stop/restart)</td>
</tr>
<tr>
<td>4</td>
<td>Log management</td>
<td>Collection of system logs and business logs</td>
</tr>
<tr>
<td>5</td>
<td>Database</td>
<td>Provision of database connection information</td>
</tr>
<tr>
<td>6</td>
<td>Patch Application</td>
<td>Displays the patch list and applies patches</td>
</tr>
<tr>
<td>7</td>
<td>Access control</td>
<td>Settings to allow or prohibit access to the execution platform</td>
</tr>
<tr>
<td>8</td>
<td>Manual scaling</td>
<td>Scaling (in/out) of the Web and AP servers</td>
</tr>
<tr>
<td>9</td>
<td>Log monitor setting and email recipient setting</td>
<td>Monitors logs and sends notifications by email</td>
</tr>
</tbody>
</table>
You can select the application execution environment definition (system configuration package) from the K5 Portal catalog according to conditions such as size and reliability.

A system configuration package allows you to **easily build and instantly use** an application execution platform.

Each system configuration package supports auto-scaling.

Establishment of system configuration, performance, and reliability requirements

Selection and deployment of a system configuration package that meets requirements

Instant access and use
System configuration packages are integrated packages for building execution platforms for applications that include environment definitions for web/application/database servers and load balancers. The packages have been verified for robustness and operability, eliminating the need for customers to design the environment themselves.

System configuration package features

<table>
<thead>
<tr>
<th>Category</th>
<th>System configuration packages address the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance &amp; scalability</td>
<td>• Preparation of system configuration tailored to system size</td>
</tr>
<tr>
<td></td>
<td>• Auto-scaling support (in future)</td>
</tr>
<tr>
<td>Reliability</td>
<td>• The ability to distribute a system across multiple availability zones (in future)</td>
</tr>
<tr>
<td></td>
<td>• Database redundancy (in future)</td>
</tr>
<tr>
<td></td>
<td>• Disaster readiness through multi-regional configuration (in future)</td>
</tr>
<tr>
<td>Security</td>
<td>• Fujitsu has completed Security audit</td>
</tr>
<tr>
<td></td>
<td>• Architecture logically separated from other customers</td>
</tr>
<tr>
<td></td>
<td>• Access control has been set via a security group</td>
</tr>
<tr>
<td></td>
<td>• Ability to select an IDS/IPS service (in future)</td>
</tr>
<tr>
<td>External connection</td>
<td>• Mechanism for communicating with multiple availability zones/regions or other sites (internet connection, IPsec VPN connection, leased line connection) (in future)</td>
</tr>
<tr>
<td>Operation &amp; maintenance</td>
<td>• Application monitoring and operation</td>
</tr>
<tr>
<td></td>
<td>• Collection of system logs and application logs</td>
</tr>
<tr>
<td></td>
<td>• Listing/Applying patches</td>
</tr>
<tr>
<td></td>
<td>• Ability to implement blue-green deployments (in future)</td>
</tr>
</tbody>
</table>
New registration, deployment, and storage history for application resources (WAR files) is simple, reducing the load on administrators and operators.

Registered resources can be deployed at the touch of a button. Generation management is based on storage history, allowing instant resource rollback.

By considering downtime at resource deployment time, the deployment group setting allows web/application server restarts to be scheduled.
A dashboard is used to display the operational status of the application execution platform in real time. Icons allow the user to **visually and easily determine** the overall operational status of the application execution platform.

- Web and application servers can be started, stopped, or restarted.
Log Management

- Administrators and users can collect system logs(*), application logs (business logs), and database logs without regard to the type or number of servers.
- System logs and business logs are automatically backed up on a daily basis.
- The current day’s system logs and business logs, as well as previously backed up system logs and business logs, can each be downloaded in ZIP format for browsing.
- Database logs can be viewed via a browser.

Example: Log browsing  WEB/AP log screen

![Image of log collection and download process]
Provides a database service that uses PostgreSQL. **Databases can be manipulated from a console.**

There is no need for managers and operators to build databases; they need only use the schemas provided to create tables, allowing databases to be used immediately.

A relay server is deployed, taking into account the security implications of database access.

---

**Database information**

1. Gathering of information

2. Database access

**Application execution platform**

- WAR
- [Business]
- [Middleware]
- [OS]
- ELB
- [Web]
- [LB]
- [AP]
- [RDB]

**SSH key file for authentication**
Patch Application

- Software and other patches from the Applicable Patches list can be used.
- Administrators and users can apply patches from the **predefined patch set** on-screen, without having to consider each individual patch.
- Patches can be applied from the screen at a time that suits the customer.
Access Control

- Provides a firewall-based access control function.
- The administrator is able to increase the security of the environment by controlling user access to web servers, application servers and jump servers.
- Access control settings can be easily registered on-screen.

Note: Access to web servers, application servers and jump servers is controlled based on originating IP addresses.
Manual Scaling

- Supports a scaling function to manually increase (decrease) web or AP servers
- Instances can be increased or decreased according to the load status
- Instances can be increased or decreased from the screen

**Normal load period**

Operates according to the number of instances for the selected system configuration package

**Peak load period**

Manually increases the number of instances according to load status, reducing the number of instances under light load conditions
Log Monitor Setting and Email Recipient Setting

- Registers and monitors keywords to be searched in log files being output to the Web server/AP server
- Notifies the registered recipient by email when a keyword is found

Note: To register email recipients, they must be registered as users on the K5 portal window. For details on how to register, refer to “K5 Portal Users Guide”.
Web API

A Web API function*1 is provided to manage the application execution platform. Cooperating with CI tools*2 used by customers allows the automation of user resource registration and deployment. Furthermore, it is possible to obtain AP logs regularly and automatically by developing application using the API.

*1 For a list of provided Web APIs, please refer to K5 Portal>Documentation>Manuals>PF>Web-API.
*2 Tools that support Continuous Integration.
**Explanation of Billing Model**

**Billing using *Fixed monthly fee + Pay-per-use***

- **Fixed monthly fee**
  
  This is a monthly fee fixed according to the individual contents of the system configuration package selected at the time the application implementation platform is built.

- **Pay-per-use**
  
  If the number of instances in the chosen system configuration package is exceeded as a result of manual scaling, this is billed as *excess instance runtime x instance count*.

  Runtime is calculated in hourly units, rounded up to the next whole number.

  For example, a run time of 1 hour 45 minutes is rounded up to 2 hours.

---

**Example:**

<table>
<thead>
<tr>
<th>8:00</th>
<th>9:00</th>
<th>10:00</th>
<th>11:00</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed monthly fee</td>
<td>Fixed monthly fee</td>
<td>Fixed monthly fee</td>
<td>Fixed monthly fee</td>
</tr>
<tr>
<td>Web</td>
<td>Web</td>
<td>Web</td>
<td>Web</td>
</tr>
<tr>
<td>AP</td>
<td>AP</td>
<td>AP</td>
<td>AP</td>
</tr>
</tbody>
</table>

Pay-per-use

- **1 hour x 1 instance**
- **1 hour x 3 instances**

**Note:** The diagram illustrates the allocation of instances over time, with specific instances marked for billing purposes.
Restrictions and Notes

- The following functions will be provided in due course:
  - Deployment group settings in User Resource Management

- The medium and large System Configuration Package models and the database redundancy model will be provided in due course.
  Note: See “Attachment: System Configuration Package List”.

- Refer to the Service Description on FUJITSU Cloud Service K5 Website to confirm the regions in which this service is offered.

- A Client ID is required to use the Web APIs and can be obtained via the PF Service application screen. When using Web APIs in an environment in which the application process has been completed, a Client ID can be obtained via the user service screen on the K5 Portal.

- The time required from application to start of service is as follows:
  - Within two business days from completing the application via the service settings application screen on the K5 Portal.
The system configuration packages provided by Fujitsu are:

<table>
<thead>
<tr>
<th>Max. no. of instances at scale-out (including initial instances)</th>
<th>Minimum scale: 5 Webs and 5 APs</th>
<th>Small: 10 Webs and 10 APs</th>
</tr>
</thead>
</table>

#### Web-Application-DB

- **CPU:16/memory:128GB**
  - **CPU:4/memory:32GB**
  - **CPU:8/memory:64GB**
  - **CPU:12/memory:96GB**
  - **CPU:16/memory:128GB**

#### Web-Application-DB (Database redundancy)

- **CPU:2/memory:8GB**
  - Mini
  - **CPU:2/memory:16GB**
  - **CPU:4/memory:32GB**
  - **CPU:8/memory:64GB**
  - **CPU:10/memory:72GB**
  - **CPU:16/memory:128GB**

#### Application-DB

- **CPU:1/memory:4GB**
  - Mini
  - **CPU:2/memory:8GB**
  - **CPU:4/memory:16GB**
  - **CPU:8/memory:32GB**
  - **CPU:10/memory:40GB**
  - **CPU:16/memory:64GB**

#### Application-DB (Database redundancy)

- **CPU:1/memory:4GB**
  - Mini
  - **CPU:2/memory:8GB**
  - **CPU:4/memory:16GB**
  - **CPU:8/memory:32GB**
  - **CPU:10/memory:40GB**
  - **CPU:16/memory:64GB**

---

**Legend**

- **Current Model**
- **Future Model**
shaping tomorrow with you