СИСТЕМА УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ СОВРЕМЕННОГО СКЛАДА С ИСПОЛЬЗОВАНИЕМ РОБОТОТЕХНИЧЕСКОГО И СЛОЖНОГО ТЕХНОЛОГИЧЕСКОГО ОБОРУДОВАНИЯ **LOGAREON WCS**

РУКОВОДСТВО ПО УСТАНОВКЕ

Москва, 2021

Оглавление

1 Введение

Документ содержит описание состава компонент системы и порядок запуска компонент.

2 Состав компонент решения WCS и принципы настройки

Для установки WCS необходимо установить следующие программные продукты:

- Redis (рекомендуется версия Redis-x64-3.0.504);
- NATS (версия v0.10.2);
- .NET Core (версия 2.2 SDK (v2.2.103) - Windows x64);
- Microsoft SQL Server 2014 или выше или PostgreSQL версии 9.5 или выше;

Конфигурация WCS состоит из следующего набора сервисов:

- **ClusterController** – кластер, управляющий всеми сервисами;
- **DataManager** – сервис для работы с базой данных;
- **IndicatorService** – сервис индикаторов, служит для оперативного мониторинга складских запасов и планирования пополнения по min\max;
- **LoggingService** – сервис логирования, используется для агрегации логов от всех сервисов с определенными фильтрами;
- **ERPIntegrationService** – сервис для интеграции с внешними системами;
- **RequirementHub** – сервис для работы с потребностями (осуществляет планирование задач на основании потребности по заказам);
- **UIService** – сервис для работы пользовательского интерфейса;
- **WHEventService** – сервис для обработки складских событий и обработчиков, позволяющих формировать задачи и документы в режиме on-line в зависимости от настроенных подписок.

Сервисы могут быть установлены как на одном сервере, так и на разных. При этом если функционал сервиса не требуется на текущей инсталляции, то устанавливать и запускать его не обязательно.

3 Порядок запуска компонент

Для установки WCS необходимо выполнить следующие действия:

1. Запустить nats-streaming-server.exe:

```
C:\Axelot_E5\Deploy\nats-streaming-server-v0.10.2-windows-386\nats-streaming-server.exe
[7892] 2020/09/13 16:13:45.714939 [32mINF[0m] STREAM: Starting nats-streaming-server[test-cluster] version 0.10.2
[7892] 2020/09/13 16:13:45.787131 [32mINF[0m] STREAM: ServerID: KElXcxoc041pRsSxkd5FmH
[7892] 2020/09/13 16:13:45.787131 [32mINF[0m] STREAM: Go version: go1.10.3
[7892] 2020/09/13 16:13:45.787131 [32mINF[0m] Starting nats-server version 1.1.0
[7892] 2020/09/13 16:13:45.788130 [32mINF[0m] Git commit [not set]
[7892] 2020/09/13 16:13:45.789130 [32mINF[0m] Listening for client connections on 0.0.0.0:4222
[7892] 2020/09/13 16:13:45.790130 [32mINF[0m] Server is ready
[7892] 2020/09/13 16:13:45.831563 [32mINF[0m] STREAM: Recovering the state...
[7892] 2020/09/13 16:13:45.832563 [32mINF[0m] STREAM: No recovered state
[7892] 2020/09/13 16:13:46.083835 [32mINF[0m] STREAM: Message store is MEMORY
[7892] 2020/09/13 16:13:46.084833 [32mINF[0m] STREAM: ---------- Store Limits ----------
[7892] 2020/09/13 16:13:46.085830 [32mINF[0m] STREAM: Channels:                  100 *
[7892] 2020/09/13 16:13:46.085830 [32mINF[0m] STREAM: --------- Channels Limits --------
[7892] 2020/09/13 16:13:46.085830 [32mINF[0m] STREAM:   Subscriptions:          1000 *
[7892] 2020/09/13 16:13:46.085830 [32mINF[0m] STREAM:   Messages     :       1000000 *
[7892] 2020/09/13 16:13:46.086828 [32mINF[0m] STREAM:   Bytes        :     976.56 MB *
[7892] 2020/09/13 16:13:46.086828 [32mINF[0m] STREAM:   Age          :     unlimited *
[7892] 2020/09/13 16:13:46.086828 [32mINF[0m] STREAM:   Inactivity   :     unlimited *
[7892] 2020/09/13 16:13:46.086828 [32mINF[0m] STREAM: ----------------------------------
[7892] 2020/09/13 16:20:16.310910 [32mINF[0m] STREAM: Channel "MainStream" has been created
[7892] 2020/09/13 16:20:16.318601 [32mINF[0m] STREAM: Channel "BadReceiptedStream" has been created
[7892] 2020/09/13 16:20:16.320599 [32mINF[0m] STREAM: Channel "RequirementChangeStream" has been created
[7892] 2020/09/13 16:20:16.321776 [32mINF[0m] STREAM: Channel "DocumentChangeStream" has been created
[7892] 2020/09/13 16:20:16.324136 [32mINF[0m] STREAM: Channel "ObjectStateStream" has been created
[7892] 2020/09/13 16:20:16.325615 [32mINF[0m] STREAM: Channel "StockChangeStream" has been created
[7892] 2020/09/13 16:29:05.192137 [32mINF[0m] STREAM: Channel "DictionaryStream" has been created
[7892] 2020/09/14 08:44:21.024359 [32mINF[0m] STREAM: Channel "TaskEvent" has been created
```

2. Запустить redis-server.exe:

```
C:\Axelot_E5\Deploy\Redis-x64-3.0.504\redis-server.exe
[8568] 13 Sep 16:13:07.107 # Warning: no config file specified, using the default config. In order to specify a config f
ile use C:\Axelot_E5\Deploy\Redis-x64-3.0.504\redis-server.exe /path/to/redis.conf

                Redis 3.0.504 (00000000/0) 64 bit

                Running in standalone mode
                Port: 6379
                PID: 8568

                      http://redis.io

[8568] 13 Sep 16:13:07.123 # Server started, Redis version 3.0.504
[8568] 13 Sep 16:13:25.231 * DB loaded from disk: 18.113 seconds
[8568] 13 Sep 16:13:25.231 * The server is now ready to accept connections on port 6379
[8568] 13 Sep 16:13:36.771 * DB saved on disk
[8568] 13 Sep 16:18:37.094 * 100 changes in 300 seconds. Saving...
[8568] 13 Sep 16:18:37.140 * Background saving started by pid 9160
[8568] 13 Sep 16:18:37.579 # fork operation complete
[8568] 13 Sep 16:18:37.579 * Background saving terminated with success
[8568] 13 Sep 16:19:49.282 * 10000 changes in 60 seconds. Saving...
[8568] 13 Sep 16:19:49.329 * Background saving started by pid 5484
```

3. Запустить redis-cli.exe. Выполнить команду ping для проверки работоспособности, при успешной работе вернётся ответ pong. Далее выполнить команду flushall для очистки кэша (ее необходимо запускать после обновления или восстановления БД):

```
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> flushall
OK
(0.75s)
127.0.0.1:6379>
```

4. Далее необходимо настроить компоненты WCS. Настройки каждого сервиса хранятся в файле appsettings.json (расположен в директории \Config), в нем необходимо указать ip

сервера, на котором запущен сервис (по-умолчанию задан localhost). Это необходимо для доступа к сервису с других компьютеров сети.

5. В файле clustermap.json необходимо указать параметры подключения к кластеру

```
{
 "ClusterInfo":
 {
  "CheckConnectionToClusterPeriodSec": 15,
  "CheckClusterStatePeriodSec": 5,
  "AuthCredentials":
  {
   "Username": "axelot",
   "Password": "axelot"
  },
  "ClusterMap":
  [
   {
    "BaseComponentURL": "http://192.168.1.4:6000/api",
    "ComponentManagerURL": "http://192.168.1.4:6000/api/manager",
    "ClusterLevel": 0
   }
  ]
 }
}
```

6. При работе с MS SQL

В файле DataManager\Config\dependencies.json выполнить настройку

```
{
  "interface": "WMS5.DataManagerBase.DAO.DB.IDatabaseManager, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL.DataManagerMSSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
{
  "interface": "WMS5.DataManagerBase.DAO.DB.IMigrateCtrl, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL.DataManagerMSSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
{
  "interface": "WMS5.DataManagerBase.Services.IWarmUpManager, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL.DataManagerMSSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
```

В конфигурационном файле DataManager\Config\datamanagerMSSQL.json необходимо указать параметры подключения к БД:

```
{
 "DataManagerMSSQLConfiguration": {
  "schemaName": "dbo",
  "DefaultIsolationLevel": "ReadCommitted",
  "ConnectionString": "Data Source=192.168.1.3;Initial Catalog=WMS5;User ID=sa;Password=1533:linliN;Connect
Timeout=30;TrustServerCertificate=True;ApplicationIntent=ReadWrite;MultiSubnetFailover=False",
  "BulkInsertParallelism": 4,
  "WriteLog": false
 }
}
```

Data Source – путь к серверу БД, Catalog – имя БД, ID – пользователь, Password – пароль.

7. При работе с MS PostgreSQL

В файле DataManager\Config\dependencies.json выполнить настройку:

```
{
  "interface": "WMS5.DataManagerBase.DAO.DB.IDatabaseManager, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL. DataManagerPostgreSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
{
  "interface": "WMS5.DataManagerBase.DAO.DB.IMigrateCtrl, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL. DataManagerPostgreSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
{
  "interface": "WMS5.DataManagerBase.Services.IWarmUpManager, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "implementation": "WMS5.DataManagerBase.Services.DB.MSSQL. DataManagerPostgreSQL, DataManagerBase, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null",
  "lifetime": "singleton"
}
```

В конфигурационном файле DataManager\Config\ datamanagerDB.json необходимо указать параметры подключения к БД:

"ConnectionString": "Server=адрес_сервера;Port=порт;User Id=имя_пользователя_бд;Password=пароль;Database=название_БД;"

На этом базовая настройка завершена и необходимо приступить к запуску сервисов. Сперва необходимо запустить ClusterController, затем DataManager, затем остальные используемые сервисы в произвольном порядке (при использовании PrintManager необходимо также запустить PrintFRServer.exe).

Сервисы можно запускать как windows service либо как обычное консольное приложение, далее рассмотрим запуск сервисов в режиме консольных приложений.

Для запуска сервиса необходимо запустить его .dll следующей командой:

dotnet ./путь к сборке/DataManager.dll –console

Например: dotnet "C:\Axelot_E5\Deploy\1581\WMS5\UIService\UIService.dll" –console

При успешном старте будет выведено соответствующее подтверждение:

```
C:\Axelot_E5\Deploy\1581\WMS5\UIService>dotnet UIService.dll -console
Hosting environment: Production
Content root path: C:\Axelot_E5\Deploy\1581\WMS5\UIService\
Now listening on: http://192.168.1.4:6101
```

Для запуска сервиса в режиме службы необходимо создать исполняемый файл (.exe). Для этого надо опубликовать проект с помощью dotnet publish в win-x64.

dotnet publish PrintManager -c Release -r win-x64

После публикации необходимо создать для него службу с помощью инструмента sc.exe.

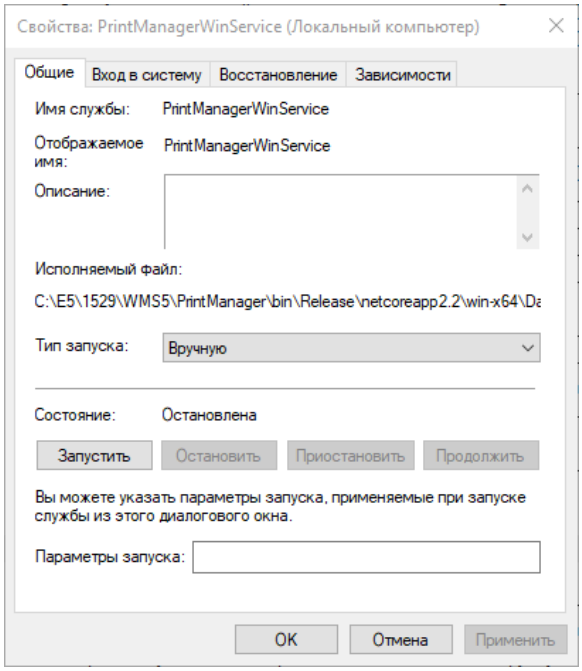
Пример создания сервиса:

sc create PrintManagerWinService binpath= C:\E5\1529\WMS5\PrintManager\bin\Release\netcoreapp2.2\win-x64\DataMartManager.exe

```
C:\WINDOWS\system32>sc create PrintManagerWinService binpath= C:\E5\1529\WMS5\PrintManager\bin\Release\netcoreapp2.2\win
-x64\DataMartManager.exe
[SC] CreateService: ycnex

C:\WINDOWS\system32>
```

Проверить что сервис корректно создался можно в services.msc:



Отсюда же сервис можно и запустить соответствующей кнопкой, после чего он перейдет в состояние «Выполняется».