First of all, thanks to be member of the BMF Community.

Please, feel free to contact us at aguerrieri@deis.unical.it for any question.

Requirements:
TinyOS 2.1.2 installed.
Java 7 installed (for the java application).

PACKAGE
In the BMF dir you can find 3 items:

- JavaApp → the java application built on top of the BMF framework (executable only);
- BMFJavaProject → the java code (that's an eclipse workspace)
- *BMF_TinyOS* folder → here there are all the TinyOS files. Put this folder wherever in your hard drive.

TINYOS

For this part of the tutorial, the UBUNTU distribution on Linux has been used. Any other Linux distribution should be supported, but it is not tested.

Once you have the last BMF version (TinyOS), you can extract it and move the *BMF_TinyOS* folder (you can rename it, but here we will always refer to *BMF_TinyOS* folder) wherever in your system.

Go in the path `/BMF_TinyOS/apps/`
Here you have the *Makerules* file. You should open it and modify the "BUILDING_MANAGEMENT_DIR" entry with the complete path where your BMF_TinyOS is.

Go in the path `/BMF_TinyOS/apps/BMF/`.

Here you can find the application files. To compile the BMF you can use a line like this:

(1) **make <PLATFORM> SENSORBOARD=<SENSORBOARD> [CC2420_CHANNEL=<CHANNEL>] [DEBUG=<DEBUG>] [MAX_GROUPS=<GROUPS>] [MAX_REQUESTS=<REQUESTS>] [ENERGY_MONITORING=<ENERGY_MONITORING>] [DISSEMINATION=<DISSEMINATION>]**

The [ ] fields are optional.

- `<PLATFORM>` is the platform you need to compile for. Right now, platforms supported are:
  - ✓ `telosb`
  - ✓ `Tyndall25`
  - ✓ `epic`
- `<SENSORBOARD>` is the sensorboard plugged on the node. The node can have no sensorboard. In this case, the sensorboard name is "`no_sensors`". Sensorboard supported are:
  - ✓ `no_sensors` → no sensors plugged on the current `<PLATFORM>` node;
  - ✓ `std_telosb` → is the standard TelosB sensorboard (light, temperature and humidity sensors)
  - ✓ `SBT80_telosb` →is the EasySen SBT80 - Multi Modality sensor board. This sensorboard is designed for TelosB. Sensors on it are: visual light, infrared, acoustic, temperature, dual-axis magnetometer and dual-axis accelerometer. (http://www.easysen.com/SBT80.htm)

- ✓ ACmeX2 →is the sensorboard used for the "BlackPlug" to sense the electricity power. It is matched to "epic" platform. (http://www.cs.berkeley.edu/~prabal/projects/epic/)
- ✓ tynsensorbrd →is one of the "Tyndall25" sensorboard. It has Light, Thermistor, Humidity, Temperature, Accelerometer and Microphone sensors.

- <CHANNEL> is the channel on which we want to transmit. Any node in the same network has to share this parameter. <CHANNEL> can have value from 11 to 26. The default value is 11.
- <DEBUG> is the Debug level you want while you'll execute the BMF on the nodes. Debug uses the serial port of the nodes to print some strings. (To be able to read the printout from the node, the node has to be plugged to a PC and the PC has to execute the application "java net.tinyos.tools.PrintfClient -comm serial@SERIAL_PORT:NODE_SPEED"). This parameter can assume these values:
  - ✓ OK → The debug is on.
  - ✓ VERBOSE → The debug is on and the prints are more detailed than in the OK version. This option is very expensive in terms of used memory.
  - ✓ NO → This is the default value. In this case, prints are off. This is the right value for the Basestation node, which uses the serial port to communicate with the BMF java API.
- <GROUPS> is the number of groups a node can belong to. This parameter can be set to a small value to save memory (this memory can be used to store requests). The default value is 10.
- <REQUESTS> is the number of the periodic requests a node can store. This parameter can be set to a small value to save memory. The default value is 10.
- <ENERGY_MONITORING> can be activated or deactivated to control the energy spent at runtime. The component for the energy spent is linked if <ENERGY_MONITORING> is set to "YES"; if <ENERGY_MONITORING> is set to "NO" the component is not linked and memory is saved. Once <ENERGY_MONITORING> is set to "YES", the user can know the time a node has the radio switched on sending data, switched on waiting for data, switched off. These requests are like sensor readings.
- <DISSEMINATION> allows the user to select which dissemination protocol has to be used running the network. Possible options are:
  - ✓ DIP → for the DIP dissemination protocol;
  - ✓ DRIP → for the Drip dissemination protocol.

  Drip is a simple, basic implementation that uses Trickle definition while DIP uses a more complex approach involving hash trees, such that it is faster, especially in packet-loss free networks. This complexity allows DIP to be significantly more efficient than existing dissemination approaches at the cost of using more program memory.

For example, to compile for telosb, a line like this can be typed:

(2) **make telosb SENSORBOARD=no_sensors CC2420_CHANNEL=20 DEBUG=NO MAX_GROUPS=10 MAX_REQUESTS=10 ENERGY_MONITORING=YES DISSEMINATION=DRIP**

To deploy the code on the nodes, a line like (1) or (2) can be used. To these lines has to be added only the normal install strings.

Example for telosb:

(3) **make telosb SENSORBOARD=SBT80_telosb install,ID_NODE CC2420_CHANNEL=20 DEBUG=NO MAX_GROUPS=10 MAX_REQUESTS=10 ENERGY_MONITORING=YES DISSEMINATION=DRIP**

Example for Tyndall25:

(4) **make Tyndall25 SENSORBOARD=tynsensorbrd install,ID_NODE CC2420_CHANNEL=22 DEBUG=NO MAX_GROUPS=10 MAX_REQUESTS=10 Tynprogbrd,SERIAL ENERGY_MONITORING=YES DISSEMINATION=DRIP**

Example for ACmeX2 node:

(5) **make epic SENSORBOARD=ACmeX2 install,ID_NODE CC2420_CHANNEL=22 DEBUG=NO MAX_GROUPS=10 MAX_REQUESTS=10 miniprog ENERGY_MONITORING=YES DISSEMINATION=DRIP**

In (3), (4), (5) ID_NODE is the ID we want to set on the node. The Basestation has to have the ID=0.

In (4) SERIAL is the serial port where the Tyndall25 programming board is plugged.

Examples of compiling or deploying strings are in the `COMPILE` and `DEPLOY` files in the `/BMF_TinyOS/apps/2012-02/` path.
Note that, in order to deploy code on Tyndall25 or ACmeX2 node, Tyndall25 and epic stacks have to be installed.

Be careful, to compile the code provided you should have all the rights on the BMF folder. To change the rights to be able to modify and compile code in the TinyOS folder, you should execute:
`sudo chmod -R 777 /opt/tinyos-2.1.2/`

JAVA

Go in a shell to execute the jars you have in the `JavaApp` folder and type:

(6) **java –jar BMF.jar SERIALPORT SPEED [MAPFILE]**

where `SERIALPORT` is the serial where the basestation is connected; `SPEED` is the basestation speed; `MAPFILE`, optional, is the address of a file to use in the "map" place in the application.

The **`SERIALPORT SPEED`** can be inserted by changing the `BMFconfiguration.cfg` file.

SENSOR SUPPORTED:

PLATFORM	Operating System	Sensors Supported
TelosB	TinyOS	Standard TelosB temperature, humidity, light sensors
TelosB	TinyOS	IR Sensor on top of the Wieye IR sensor board
Tyndall25	TinyOS	Tyndall accelerometer, light, sound, temperature, magnetic fields sensors
Epic	TinyOS	ACme Electricity sensor
SunSPOT	Squawk VM	Accelerometer, light and temperature SunSPOT sensors

Note: To use the java-side part of any application that wants to interface with the TinyOS API, you should run the application on a 32 bit Java distribution.

TYNYOS: HOW TO ADD A NEW PLATFORM TO THE BMF

This example will show how the Shimmer platform has been added to the BMF.
Consideration: Shimmer nodes use the same radio and microcontroller than the TelosB nodes.

Trying to compile the BMF for Shimmer2r with the following line:

```
make shimmer2r SENSORBOARD=no_sensors DEBUG=VERBOSE MAX_GROUPS=5
MAX_REQUESTS=5 CC2420_CHANNEL=18 ENERGY_MONITORING=NO DISSEMINATION=DRIP
```

This is the result:

```
CommManagerC.nc:79:3: error: #error "BMF is only supported for mica2,
micaz, telos, tyndall25, epic and aquisgrain nodes"
In component `CommManagerC':
CommManagerC.nc:82: cannot find `LPLProvider'
```

So, the first component to change is the CommManagerC.nc
In the "if" "`// RADIO STUFF for LOW POWER`" for the precompiler add the piece of code:

```
|| defined(PLATFORM_SHIMMER2R)
```

That means that we are considering the Shimmer platform and we are linking the `CC2420ActiveMessageC` component as the radio controller.
If a new platform to be added supports the low power listening interface, it can be added here.

Go to `/BMF_TinyOS/support/make/` folder and modify the `buildingManagement.extra` file.
In particular, add the following line to the #SENSOR BOARD SELECTION section.

```
else ifeq ($(SENSORBOARD),shimmer2r_kinematics)
   PFLAGS += -DSENSORBOARD=6
```

This is to have the possibility to understand everywhere in the TinyOS code which sensors has the current node. `shimmer2r_kinematics` sensorboard is the one providing accelerometer and gyroscope (gyro is not actually supported in the BMF).

In `/BMF_TinyOS/tos/platforms/` copy the `telosb` folder in a `shimmer2r` folder .

In `/BMF_TinyOS/tos/sensorboards/` copy the `no_sensors` folder in a `shimmer2r_kinematics` folder . Now, in the `sensorAbstractions` folder, the interested sensor abstractions can be redefined and the other files can be deleted.

In `/BMF_TinyOS/tos/YOUR_APP_VERSION/` open the file `BMF.h` and add the line:

```
SENSORBOARD_TYPE_SHIMMER2R_KINEMATICS =6,
```

Be careful, the constant must be the same that is set in the `buildingManagement.extra` file.

`ENERGY_MONITORING=YES` is not supported for Shimmer node. Leave it to `ENERGY_MONITORING=NO`.

Now, once a Shimmer node is plugged to your PC, it can be seen through the command:

```
ls -l /dev/ttyUSB*
```

Before deploying the TinyOS code on the Shimmer, authorizations must be changed:

```
sudo chmod 666 /dev/ttyUSB0
```

where `/dev/ttyUSB0` is the COM port for the Shimmer node.
Finally, to deploy the code:

```
make shimmer2r install bsl,/dev/ttyUSB0
```

So, to deploy the actual BMF version, the following command can be written:

```
make shimmer2r SENSORBOARD=shimmer2r_kinematics install,1
CC2420_CHANNEL=20 DEBUG=NO MAX_GROUPS=10 MAX_REQUESTS=10
ENERGY_MONITORING=NO DISSEMINATION=DRIP FUNCTIONS=STANDARD
bsl,/dev/ttyUSB0
```

Here, the ID for the Shimmer node is 1.

JAVA: HOW TO ADD A NEW PLATFORM TO THE BMF

Add the constants to the Constants.java file (package ie.ucd.clarity.bmf.common):

```
static final public int SENSORBOARD_TYPE_SHIMMER2R_KINEMATICS    =6;

static final public String
SENSORBOARD_TYPE_SHIMMER2R_KINEMATICS_DESCRIPTION     = "Sensorboard
Kinematics on Shimmer node.\n"  + "   SENSORS ONBOARD: \n" + "      -
Accelerometer;\n" + "      - Gyroscope.";
```

In the TinyOSBMFManager (package ie.ucd.clarity.bmf.network.platform.tinyOS), in the setup() method
Update the lines:
this.sensorBoardsNames = new ArrayList<String>(7);
this.sensors = new int [7][];
with the new number of sensorboards;

add the line:

```
this.sensorBoardsNames.add("Sensorboard Kinematics on Shimmer node");
```